N 15

```
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMMMMM   MMMMMM  000         000  MMMMMM   MMMMMM
MMMMMM   MMMMMM  000         000  MMMMMM   MMMMMM
MMMMMM   MMMMMM  000         000  MMMMMM   MMMMMM
MMM   MMM   MMM  000         000  MMM   MMM   MMM
MMM   MMM   MMM  000         000  MMM   MMM   MMM
MMM   MMM   MMM  000         000  MMM   MMM   MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM  000         000  MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
MMM         MMM     000000000     MMM         MMM
```

```
**FILE**ID**MOMTEST                                   G  1




MM      MM    000000    MM      MM  TTTTTTTTTT  EEEEEEEEEE    SSSSSSSS  TTTTTTTTTT
MM      MM    000000    MM      MM  TTTTTTTTTT  EEEEEEEEEE    SSSSSSSS  TTTTTTTTTT
MMMM  MMMM  OO      OO  MMMM  MMMM      TT      EE          SS              TT
MMMM  MMMM  OO      OO  MMMM  MMMM      TT      EE          SS              TT
MM  MM  MM  OO      OO  MM  MM  MM      TT      EE          SS              TT
MM  MM  MM  OO      OO  MM  MM  MM      TT      EE          SS              TT
MM      MM  OO      OO  MM      MM      TT      EEEEEEEE      SSSSSS        TT
MM      MM  OO      OO  MM      MM      TT      EEEEEEEE      SSSSSS        TT
MM      MM  OO      OO  MM      MM      TT      EE                  SS      TT
MM      MM  OO      OO  MM      MM      TT      EE                  SS      TT
MM      MM  OO      OO  MM      MM      TT      EE                  SS      TT      ....
MM      MM  OO      OO  MM      MM      TT      EE                  SS      TT      ....
MM      MM    000000    MM      MM      TT      EEEEEEEEEE  SSSSSSSS        TT      ....
MM      MM    000000    MM      MM      TT      EEEEEEEEEE  SSSSSSSS        TT      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0001   0 %TITLE 'MOM Loop Test Routines'
0002   0 MODULE MOMTEST (
0003   0                 LANGUAGE (BLISS32),
0004   0                 ADDRESSING_MODE (NONEXTERNAL=GENERAL),
0005   0                 ADDRESSING_MODE (EXTERNAL=GENERAL),
0006   0                 IDENT = 'V04-000'
0007   0                 ) =
0008   1 BEGIN
0009   1 !
0010   1 !*****************************************************************************
0011   1 !*                                                                           *
0012   1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0013   1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0014   1 !*  ALL RIGHTS RESERVED.                                                     *
0015   1 !*                                                                           *
0016   1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0017   1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0018   1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0019   1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0020   1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0021   1 !*  TRANSFERRED.                                                             *
0022   1 !*                                                                           *
0023   1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0024   1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0025   1 !*  CORPORATION.                                                             *
0026   1 !*                                                                           *
0027   1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0028   1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0029   1 !*                                                                           *
0030   1 !*                                                                           *
0031   1 !*****************************************************************************
0032   1 !
0033   1
0034   1 !++
0035   1 ! FACILITY:  DECnet-VAX Network Management Maintenance Operations Module (MOM)
0036   1 !
0037   1 ! ABSTRACT:
0038   1 !
0039   1 !       This routine contains routines to process NCP LOOP NODE and LINE
0040   1 !       command messages.
0041   1 !
0042   1 ! ENVIRONMENT:  VAX/VMS Operating System
0043   1 !
0044   1 ! AUTHOR:  Kathy Perko
0045   1 !
0046   1 ! CREATION DATE:  9-Jan-1983
0047   1 !
0048   1 ! MODIFIED BY:
0049   1 !       V03-006 MKP0006         Kathy Perko             22-July-1984
0050   1 !               Fix LOOP CIRCUIT on point-to-point circuits so that,
0051   1 !               after getting the first invalid OC message (to synchronize
0052   1 !               with the target), to skip over all subsequent OC messages.
0053   1 !
0054   1 !       V03-005 MKP0005         Kathy Perko             5-June-1984
0055   1 !               Allow LOOP NODE to work if the node is specified by number.
0056   1 !
0057   1 !       V03-004 MKP0004         Kathy Perko             29-April-1984
```

```
;    58         0058  1 !                    When looping a circuit in the Ethernet, make sure the
;    59         0059  1 !                    HELP NI address is not alternated by MOM$MOPSNDRCV.  This
;    60         0060  1 !                    is because the loop message should be rebuilt to match
;    61         0061  1 !                    the NI address being alternated.  That must be done outside
;    62         0062  1 !                    MOM$MOPSNDRCV.
;    63         0063  1 !
;    64         0064  1 !            V03-003 MKP0003          Kathy Perko               22-Jan-1984
;    65         0065  1 !                    Fix LOOP NODE so it works with access control.
;    66         0066  1 !                    Add function code parameter to calls to MOM$INIT_CIB.
;    67         0067  1 !
;    68         0068  1 !            V03-002 MKP0002          Kathy Perko               6-Dec-1983
;    69         0069  1 !                    If mapping a loop error I don't recognize, return whatever
;    70         0070  1 !                    Network Management completion code the caller gave me, instead
;    71         0071  1 !                    of returning "Management Program Error".
;    72         0072  1 !
;    73         0073  1 !            V03-001 MKP0001          Kathy Perko               6-May-1983
;    74         0074  1 !                    Fix DMC loop circuit.  Fix loop line.
;    75         0075  1 !
;    76         0076  1 !--
;    77         0077  1
;    79          0078   1 %SBTTL 'Declarations'
;    80          0079   1
;    81          0080   1 !
;    82          0081   1 ! TABLE OF CONTENTS:
;    83          0082   1 !
;    84          0083   1
;    85          0084   1 FORWARD ROUTINE
;    86          0085   1     mom$test             : NOVALUE,
;    87          0086   1     mom$activeloop       : NOVALUE,
;    88          0087   1     mom_build_loop_bufs  : NOVALUE,
;    89          0088   1     mom$passiveloop      : NOVALUE,
;    90          0089   1     mom$loop_line        : NOVALUE,
;    91          0090   1     mom$nodetest         : NOVALUE,
;    92          0091   1     mom$initbuffer       : NOVALUE,
;    93          0092   1     mom$chkbuffer,
;    94          0093   1     mom$getbuffer,
;    95          0094   1     mom$freebuffer,
;    96          0095   1     mom_openlink         : NOVALUE,
;    97          0096   1     mom_bldloopnfb       : NOVALUE,
;    98          0097   1     mom_maplooperr       : NOVALUE,
;    99          0098   1     mom_siglooperr       : NOVALUE,
;   100          0099   1     mom$saveuser,
;   101          0100   1     mom$savepasswrd,
;   102          0101   1     mom$saveacct,
;   103          0102   1     mom$loophandler,
;   104          0103   1     mom$testhandler;
;   105          0104   1
;   106          0105   1 !
;   107          0106   1 ! INCLUDE FILES:
;   108          0107   1 !
;   109          0108   1
;   110          0109   1 LIBRARY 'LIB$:MOMLIB.L32';
;   111          0110   1 LIBRARY 'SHRLIB$:NMALIBRY.L32';
;   112          0111   1 LIBRARY 'SHRLIB$:EVCDEF.L32';
;   113          0112   1 LIBRARY 'SHRLIB$:NET.L32';
;   114          0113   1 LIBRARY 'SYS$LIBRARY:LIB.L32';            ! Network ACP control QIO interface
;   115          0114   1
;   116          0115   1 !
;   117          0116   1 ! EQUATED SYMBOLS:
;   118          0117   1 !
;   119          0118   1
;   120          0119   1 LITERAL
;   121          0120   1     mbx_size = 40,
;   122          0121   1     nfb_bufsize = 110;
;   123          0122   1
;   124          0123   1 !
;   125          0124   1 ! OWN STORAGE:
;   126          0125   1 !
;   127          0126   1
;   128          0127   1 OWN
;   129          0128   1     loop_chan,                              ! Loop link channel
;   130          0129   1     loop_mbxchan;                           ! Loop mailbox channel
;   131          0130   1
;   132          0131   1 OWN
;   133          0132   1     userdsc     : VECTOR [2],               ! User id
;   134          0133   1     accountdsc  : VECTOR [2],               ! Account
;   135          0134   1     passworddsc : VECTOR [2];               ! Password
```

```
1 !
1 ! NFB and P2 buffers and descriptors for loop QIO.
1 !
1 OWN
1     nfbdsc      : VECTOR [2],            ! NFB for loop link
1     mbx_buffer : VECTOR [40, BYTE],      ! Mailbox buffer
1     nfb_buffer : VECTOR [nfb_bufsize, BYTE], ! NFB buffer
1     p2_buffer  : VECTOR [mom$k_p2_buf_len,  BYTE]; ! P2 QIO buffer
1
1 BIND
1     mom$q_nfb_buf_dsc  = UPLIT (%ALLOCATION(nfb_buffer),  nfb_buffer)
1                         : VECTOR [2],
1     mom$q_p2bfdsc   = UPLIT (%ALLOCATION(p2_buffer),  p2_buffer)
1                         : VECTOR [2];
1
1 !
1 ! The following are for Phase 2 and Phase 3 differences.
1 !
1 OWN
1     version;
1
1 BIND
2     object25dsc = $ASCID ('::"25=/',%CHAR(0,0,0),'                    ','"')
1                 : VECTOR [2],
2     object19dsc = $ASCID ('::"19=/',%CHAR(0,0,0),'                    ','"')
1                 : VECTOR [2];
1 !
1 ! The following data is used to manage buffers.  Default buffers may be
1 ! used or the buffers may be allocated from virtual memory.
1 !
1 LITERAL
1     mom$k_defbufsize = 130,
1     mom$k_maxmsgsize = 128;
1
1 OWN
1     mom$l_mop_chan,
1     mom$l_assist_mop_chan,
1     mom$l_vmbufsize  : LONG INITIAL (0),
1     mom$l_vm_buf_adr : LONG INITIAL (0),
1     mom$t_testrcvbuf : VECTOR [mom$k_defbufsize, BYTE];
1
1 !
1 ! EXTERNAL REFERENCES:
1 !
1
1 $mom_externals;
1
1 EXTERNAL
1     mom$gq_netnamdsc,
1     mom$gq_dle_namdsc,
1     mom$gq_psinamdsc,
1     mom$npa_test,
1     mom$npa_test_node_acc,
1     mom$npa_cirloop;
1
1 EXTERNAL LITERAL
```

```
;  193         0192  1        mom$_alpbfovf,
;  194         0193  1        mom$_mirbfovf,
;  195         0194  1        mom$_ncbfail;
;  196         0195  1
;  197         0196  1   EXTERNAL ROUTINE
;  198         0197  1        nma$nparse,
;  199         0198  1        mom$bld_reply,
;  200         0199  1        mom$build_p2,
;  201         0200  1        mom$chk_mop_error,
;  202         0201  1        mom$debug_msg,
;  203         0202  1        mom$debug_qio,
;  204         0203  1        mom$error,
;  205         0204  1        mom$get_circuit_type,
;  206         0205  1        mom$get_node_id,
;  207         0206  1        mom$getsrvtimer,
;  208         0207  1        mom$init_CIB,
;  209         0208  1        mom$log_event,
;  210         0209  1        mom$netacp_qio,
;  211         0210  1        mom$mapqioerror,
;  212         0211  1        mom$mopopen,
;  213         0212  1        mom$mopsetsubstate,
;  214         0213  1        mom$mopsndrcv,
;  215         0214  1        lib$get_vm,
;  216         0215  1        lib$free_vm,
;  217         0216  1        lib$asn_wth_mbx : ADDRESSING_MODE (GENERAL);
;  218         0217  1
```

```
1 %SBTTL 'mom$test  Loopback test'
1 GLOBAL ROUTINE mom$test : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine is called when MOM receives a NICE LOOP command.
1 !       The NICE message parsing is completed (it has already been parsed
1 !       through the circuit, line or node ID), and the routine to perform
1 !       the loop is called.
1 !
1 !--
1
2 BEGIN
2
2 MAP
2     mom$gb_option_byte: bblock [1];
2
2 LOCAL
2     nparse_table,
2     loop_routine,
2     status;
2
2 !
2 ! Parse the remainder of the NICE command to get loop parameters.
2 !
2 SELECTONEU .mom$gb_entity_code OF
2     SET
2     [mom$c_node, mom$c_nodebyname]:
3         BEGIN
3         !
3         ! If there is access control in the LOOP NODE command, use a
3         ! different parsing table to get it from the NICE command.
3         !
3         IF .mom$gl_service_flags [mom$v_loop_w_access_ctl]
3         THEN
3             nparse_table = mom$npa_test_node_acc
3         ELSE
3             nparse_table = mom$npa_test;
3         loop_routine = mom$nodetest;
2         END;
2
2     [mom$c_line]:
3         BEGIN
3         nparse_table = mom$npa_test;
3         loop_routine = mom$loop_line;
2         END;
2
2     [mom$c_circuit]:
3         BEGIN
3         nparse_table = mom$npa_cirloop;
3         loop_routine = mom$activeloop;
2         END;
2     TES;
2 !
2 ! All parsing errors will be signalled by the action routines so, if control
2 ! returns here, the parsing was successful.
2 !
```

```
;  277           0275  2 nma$nparse (mom$ab_nparse_blk, .nparse_table);
;  278           0276  2 (.loop_routine) ();
;  279           0277  1 END;                                ! End of MOM$TEST


                                                                   .TITLE  MOMTEST MOM Loop Test Routines
                                                                   .IDENT  \V04-000\

                                                                   .PSECT  $PLIT$,NOWRT,NOEXE,2

                                      0000006E  00000 P.AAA:       .LONG    110                                    ;
                                      00000000' 00004              .ADDRESS NFB_BUFFER                             ;
                                      00000068  00008 P.AAB:       .LONG    104                                    ;
                                      00000000' 0000C              .ADDRESS P2_BUFFER                              ;
20 20 20 20 20 00  00 00 2F 3D 35 32 22 3A 3A  00010 P.AAD:       .ASCII   \::"25=/\<0><0><0>\          \         ;
                   20 20 20 20 20 20 20 20 20  0001F                                                               ;
                                     22 20 20  00028              .ASCII   \  "\                                   ;
                                               0002B              .BLKB    1                                       ;
                                      0000001B  0002C P.AAC:       .LONG    27                                     ;
                                      00000000' 00030              .ADDRESS P.AAD                                  ;
20 20 20 20 20 00  00 00 2F 3D 39 31 22 3A 3A  00034 P.AAF:       .ASCII   \::"19=/\<0><0><0>\          \         ;
                   20 20 20 20 20 20 20 20 20  00043                                                               ;
                                     22 20 20  0004C              .ASCII   \  "\                                   ;
                                               0004F              .BLKB    1                                       ;
                                      0000001B  00050 P.AAE:       .LONG    27                                     ;
                                      00000000' 00054              .ADDRESS P.AAF                                  ;

                                                                   .PSECT  $OWN$,NOEXE,2

                                                00000 LOOP_CHAN:
                                                                   .BLKB    4
                                                00004 LOOP_MBXCHAN:
                                                                   .BLKB    4
                                                00008 USERDSC:.BLKB    8
                                                00010 ACCOUNTDSC:
                                                                   .BLKB    8
                                                00018 PASSWORDDSC:
                                                                   .BLKB    8
                                                00020 NFBDSC: .BLKB    8
                                                00028 MBX_BUFFER:
                                                                   .BLKB    40
                                                00050 NFB_BUFFER:
                                                                   .BLKB    110
                                                000BE              .BLKB    2
                                                000C0 P2_BUFFER:
                                                                   .BLKB    104
                                                00128 VERSION:.BLKB    4
                                                0012C MOM$L_MOP_CHAN:
                                                                   .BLKB    4
                                                00130 MOM$L_ASSIST_MOP_CHAN:
                                                                   .BLKB    4
                                      00000000  00134 MOM$L_VMBUFSIZE:
                                                                   .LONG    0                                      ;
                                      00000000  00138 MOM$L_VM_BUF_ADR:
                                                                   .LONG    0                                      ;
                                                0013C MOM$T_TESTRCVBUF:
                                                                   .BLKB    130
```

```
MOM$Q_NFB_BUF_DSC=  P.AAA
MOM$Q_P2BFDSC=      P.AAB
OBJECT25DSC=        P.AAC
OBJECT19DSC=        P.AAE
          .EXTRN   MOM$GL_LOGMASK, MOM$GL_SVD_INDEX
          .EXTRN   MOM$AB_SERVICE_DATA
          .EXTRN   MOM$GB_FUNCTION
          .EXTRN   MOM$GB_OPTION_BYTE
          .EXTRN   MOM$GB_ENTITY_CODE
          .EXTRN   MOM$AB_ENTITY_BUF
          .EXTRN   MOM$GQ_ENTITY_BUF_DSC
          .EXTRN   MOM$GL_SERVICE_FLAGS
          .EXTRN   MOM$AB_NPARSE_BLK
          .EXTRN   MOM$AB_NICE_RCV_BUF
          .EXTRN   MOM$AB_NICE_XMIT_BUF
          .EXTRN   MOM$GQ_NICE_RCV_BUF_DSC
          .EXTRN   MOM$GL_NICE_RCV_MSG_LEN
          .EXTRN   MOM$GQ_NICE_XMIT_BUF_DSC
          .EXTRN   MOM$AB_MSGBLOCK
          .EXTRN   MOM$AB_ACPQIO_BUFFER
          .EXTRN   MOM$GQ_ACPQIO_BUF_DSC
          .EXTRN   MOM$AB_CIB, MOM$AB_LOOP_CIB
          .EXTRN   MOM$AB_TRIGGER_CIB
          .EXTRN   MOM$AB_MOP_XMIT_BUF
          .EXTRN   MOM$GQ_MOP_XMIT_BUF_DSC
          .EXTRN   MOM$AB_MOP_RCV_BUF
          .EXTRN   MOM$GQ_MOP_RCV_BUF_DSC
          .EXTRN   MOM$AB_MOP_MSG, MOM$GQ_MOP_MSG_DSC
          .EXTRN   MOM$GW_EVT_CODE
          .EXTRN   MOM$GB_EVT_POPR
          .EXTRN   MOM$GB_EVT_PRSN
          .EXTRN   MOM$GB_EVT_PSER
          .EXTRN   SVD$GK_PCNO_ADD
          .EXTRN   SVD$GK_PCNO_SDV
          .EXTRN   SVD$GK_PCNO_CPU
          .EXTRN   SVD$GK_PCNO_STY
          .EXTRN   SVD$GK_PCNO_DAD
          .EXTRN   SVD$GK_PCNO_DCT
          .EXTRN   SVD$GK_PCNO_IHO
          .EXTRN   SVD$GK_PCNO_NNA
          .EXTRN   SVD$GK_PCNO_SLI
          .EXTRN   SVD$GK_PCNO_SPA
          .EXTRN   SVD$GK_PCNO_HWA
          .EXTRN   SVD$GK_PCNO_SNV
          .EXTRN   SVD$GK_PCNO_LOA
          .EXTRN   SVD$GK_PCNO_SLO
          .EXTRN   SVD$GK_PCNO_TLO
          .EXTRN   SVD$GK_PCNO_DFL
          .EXTRN   SVD$GK_PCNO_SID
          .EXTRN   SVD$GK_PCNO_DUM
          .EXTRN   SVD$GK_PCNO_SDU
          .EXTRN   SVD$GK_PCNO_$HNA
          .EXTRN   SVD$GK_PCNO_$HHW
          .EXTRN   SVD$GK_PCNO_$FTY
          .EXTRN   SVD$GK_PCNO_PHA
          .EXTRN   SVD$GK_PCNO_$DA
```

```
                                                        .EXTRN  SVD$GK_PCNO_LPC
                                                        .EXTRN  SVD$GK_PCNO_LPL
                                                        .EXTRN  SVD$GK_PCNO_LPD
                                                        .EXTRN  SVD$GK_PCNO_LPH
                                                        .EXTRN  SVD$GK_PCNO_LPA
                                                        .EXTRN  SVD$GK_PCNO_LPN
                                                        .EXTRN  SVD$GK_PCNO_$LNA
                                                        .EXTRN  SVD$GK_PCNO_$LNH
                                                        .EXTRN  SVD$GK_PCNO_LAN
                                                        .EXTRN  SVD$GK_PCNO_$LNN
                                                        .EXTRN  SVD$GK_PCNO_$LAH
                                                        .EXTRN  SVD$GK_PCLI_STI
                                                        .EXTRN  SVD$C_ENTRY_COUNT
                                                        .EXTRN  MOM$GQ_NETNAMDSC
                                                        .EXTRN  MOM$GQ_DLE_NAMDSC
                                                        .EXTRN  MOM$GQ_PSINAMDSC
                                                        .EXTRN  MOM$NPA_TEST, MOM$NPA_TEST_NODE_ACC
                                                        .EXTRN  MOM$NPA_CIRLOOP
                                                        .EXTRN  MOM$_ALPBFOVF, MOM$_MIRBFOVF
                                                        .EXTRN  MOM$_NCBFAIL, NMA$NPARSE
                                                        .EXTRN  MOM$BLD_REPLY, MOM$BUILD_P2
                                                        .EXTRN  MOM$CHK_MOP_ERROR
                                                        .EXTRN  MOM$DEBUG_MSG, MOM$DEBUG_QIO
                                                        .EXTRN  MOM$ERROR, MOM$GET_CIRCUIT_TYPE
                                                        .EXTRN  MOM$GET_NODE_ID
                                                        .EXTRN  MOM$GETSRVTIMER
                                                        .EXTRN  MOM$INIT_CIB, MOM$LOG_EVENT
                                                        .EXTRN  MOM$NETACP_QIO, MOM$MAPQIOERROR
                                                        .EXTRN  MOM$MOPOPEN, MOM$MOPSETSUBSTATE
                                                        .EXTRN  MOM$MOPSNDRCV, LIB$GET_VM
                                                        .EXTRN  LIB$FREE_VM, LIB$ASN_WTH_MBX

                                                        .PSECT  $CODE$,NOWRT,2

                                    000C 00000          .ENTRY  MOM$TEST, Save R2,R3                              ; 0219
             53 00000000G    00       9E 00002          MOVAB   MOM$NPA_TEST, R3                                  ;
             50 00000000G    00       9A 00009          MOVZBL  MOM$GB_ENTITY_CODE, R0                            ; 0243
             01                    50 91 00010          CMPB    R0, #1                                            ; 0245
                                   1D 1A 00013          BGTRU   3$                                                ;
     09 00000000G    00            04 E1 00015          BBC     #4, MOM$GL_SERVICE_FLAGS, 1$                      ; 0251
             50 00000000G    00       9E 0001D          MOVAB   MOM$NPA_TEST_NODE_ACC, NPARSE_TABLE               ; 0253
                                   03 11 00024          BRB     2$                                                ;
             50                    63 9E 00026 1$:      MOVAB   MOM$NPA_TEST, NPARSE_TABLE                        ; 0255
             52 00000000V    00       9E 00029 2$:      MOVAB   MOM$NODETEST, LOOP_ROUTINE                        ; 0256
                                   24 11 00030          BRB     5$                                                ; 0243
             03                    50 91 00032 3$:      CMPB    R0, #3                                            ; 0259
                                   0C 12 00035          BNEQ    4$                                                ;
             50                    63 9E 00037          MOVAB   MOM$NPA_TEST, NPARSE_TABLE                        ; 0261
             52 00000000V    00       9E 0003A          MOVAB   MOM$LOOP_LINE, LOOP_ROUTINE                       ; 0262
                                   13 11 00041          BRB     5$                                                ; 0243
             02                    50 91 00043 4$:      CMPB    R0, #2                                            ; 0265
                                   0E 12 00046          BNEQ    5$                                                ;
             50 00000000G    00       9E 00048          MOVAB   MOM$NPA_CIRLOOP, NPARSE_TABLE                     ; 0267
             52 00000000V    00       9E 0004F          MOVAB   MOM$ACTIVELOOP, LOOP_ROUTINE                      ; 0268
                                   50 DD 00056 5$:      PUSHL   NPARSE_TABLE                                      ; 0275
                00000000G    00       9F 00058          PUSHAB  MOM$AB_NPARSE_BLK                                 ;
        00000000G    00            02 FB 0005E          CALLS   #2, NMA$NPARSE                                    ;
```

```
                                62              00  FB 00065          CALLS   #0, (LOOP_ROUTINE)                              ; 0276
                                                    04 00068          RET                                                     ; 0277

; Routine Size:  105 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 %SBTTL 'mom$activeloop  Active loop'
1 GLOBAL ROUTINE mom$activeloop : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine performs the active loop function as a result
1 !       a LOOP CIRCUIT command.
1 !
1 ! IMPLICIT INPUTS:
1 !       MOM$L_MOP_CHAN = Channel number on which to issue loop MOP messages.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       Signal errors.
1 !
1 !--
1
2 BEGIN
2
2 OWN
2     msgs_looped;
2
2 LOCAL
2     mom_recv_CIB: REF BBLOCK,
2     mom_xmit_CIB: REF BBLOCK,
2     loop_count : WORD,
2     help_type,
2     msgsize,
2     rcvlen,
2     retry,
2     skip_invalid_MOP_msg,
2     snddsc       : VECTOR [2],
2     rcvdsc       : VECTOR [2],
2     status;
2
2 !
2 ! Enable condition handler to clean up after loop operation.
2 !
2 ENABLE mom$loophandler;
2 mom$get_circuit_type ();
2 mom$getsrvtimer ();
2 IF NOT .mom$gl_service_flags [mom$v_ni_circ] THEN
2     !
2     ! Assign a MOP channel for sending point to point or multipoint loop
2     ! messages and make sure CIB isn't treated as if the circuit were an
2     ! NI.
2     !
3     BEGIN
3     mom$mopopen (mom$ab_cib [cib$l_chan]);
3     mom$init_CIB (mom$ab_cib, nma$c_fnc_tes, 0, 0, 0);
3     mom$ab_cib [cib$v_target_addr_fixed] = true;
3     mom_xmit_CIB = mom_recv_CIB = mom$ab_cib;
3     END
2 ELSE
3     BEGIN
```

```
; 3     !
; 3     ! If the operation is LOOP with an assistant node (NI only) and there
; 3     ! was no HELP type specified in the command, default HELP type to FULL.
; 3     ! This means the looped message will go to the assisting node both on
; 3     ! the way to and on the way back from the target node.
; 3     !
; 3     IF .mom$gl_service_flags [mom$v_loop_w_assist] AND
; 3        NOT .mom$ab_service_data [svd$gk_pcno_lph, svd$v_msg_param] THEN
; 3         mom$ab_service_data [svd$gk_pcno_lph, svd$l_param] = nma$c_loop_full;
; 3     help_type = .mom$ab_service_data [svd$gk_pcno_lph, svd$l_param];
; 3     !
; 3     ! Set up to do QIOs to the target node.  First, get the target node's
; 3     ! hardware address and/or node number.  Then, get a MOP I/O channel,
; 3     ! and set up the association between the MOP I/O channel and an NI
; 3     ! destination.
; 3     !
; 3     IF NOT .mom$ab_service_data [svd$gk_pcno_pha, svd$v_msg_param] THEN
; 3         mom$get_node_id (svd$gk_pcno_lpn,
; 3                          svd$gk_pcno_$lna,
; 3                          svd$gk_pcno_$lnh);
; 3     IF .help_type NEQ nma$c_loop_full THEN
; 4         BEGIN
; 4         mom$mopopen (mom$ab_cib [cib$l_chan]);
; 4         mom$init_CIB (mom$ab_cib,
; 4                       nma$c_fnc_tes,
; 4                       svd$gk_pcno_pha,
; 4                       svd$gk_pcno_lpn,
; 4                       svd$gk_pcno_$lnh);
; 3         END;
; 3     !
; 3     ! If loop assistance is specified in the NICE command, the loop message
; 3     ! must be transmitted or received by a second node.  Set up this assistance
; 3     ! channel the same way as the target channel (above).
; 3     !
; 3     If .help_type NEQ mom$k_no_loop_help THEN
; 4         BEGIN
; 4         mom$mopopen (mom$ab_loop_cib [cib$l_chan]);
; 4         IF NOT .mom$ab_service_data [svd$gk_pcno_lpa, svd$v_msg_param] THEN
; 4             mom$get_node_id (svd$gk_pcno_lan,
; 4                              svd$gk_pcno_$lnn,
; 4                              svd$gk_pcno_$lah);
; 4         mom$init_CIB (mom$ab_loop_cib,
; 4                       nma$c_fnc_tes,
; 4                       svd$gk_pcno_lpa,
; 4                       svd$gk_pcno_lan,
; 4                       svd$gk_pcno_$lnh);
; 4         !
; 4         ! The loop message must be rebuilt to alternate with the receive
; 4         ! NI address, so tell MOM$MOPSNDRCV not alternate the receive NI
; 4         ! address.  That must be done from here (although it isn't done
; 4         ! correctly right now).
; 4         !
; 4         mom$ab_loop_cib [cib$v_target_addr_fixed] = true;
; 3         END;
; 3
; 3     SELECT .help_type of
; 3         SET
;   395        0392  3           [mom$k_no_loop_help, nma$c_loop_recv]:
;   396        0393  3               !
;   397        0394  3               ! Set the loop messages to be transmitted to the target node.
;   398        0395  3               !
;   399        0396  3               mom_xmit_cib = mom$ab_cib;
;   400        0397  3           [nma$c_loop_full, nma$c_loop_xmit]:
;   401        0398  3               !
;   402        0399  3               ! Set the loop messages to be transmitted to the assisting node.
;   403        0400  3               !
;   404        0401  3               mom_xmit_cib = mom$ab_loop_cib;
;   405        0402  3           [nma$c_loop_full, nma$c_loop_recv]:
;   406        0403  3               !
;   407        0404  3               ! Set the looped messages to be received from the assisting node.
;   408        0405  3               !
;   409        0406  3               mom_recv_cib = mom$ab_loop_cib;
;   410        0407  3           [mom$k_no_loop_help, nma$c_loop_xmit]:
;   411        0408  3               !
;   412        0409  3               ! Set the loop messages to be received from the target node.
;   413        0410  3               !
;   414        0411  3               mom_recv_cib = mom$ab_cib;
;   415        0412  3           TES;
;   416        0413  2       END;
;   417        0414  2   !
;   418        0415  2   ! Set the circuit substate to -LOOPING.
;   419        0416  2   !
;   420        0417  2   mom$mopsetsubstate (nma$c_linss_loo,
;   421        0418  2                         .mom_xmit_cib [cib$l_chan]);
;   422        0419  2   mom_build_loop_bufs (snddsc, rcvdsc);
;   423        0420  2   !
;   424        0421  2   ! Get the number of times to loop from the data base.
;   425        0422  2   ! If the parameter is not set in the data base then use default size.
;   426        0423  2   !
;   427        0424  2   loop_count = .mom$ab_service_data [svd$gk_pcno_lpc, svd$l_param];
;   428        0425  2   !
;   429        0426  2   ! Loop the specified number of times.  For point to point circuits,
;   430        0427  2   ! the target node's driver doesn't keep the MOP message that causes MOM to
;   431        0428  2   ! get start up.  Therefore, the target's MOM will send a bogus OC MOP message
;   432        0429  2   ! to get this MOM to retransmit the loop message.  After receiving the first
;   433        0430  2   ! OC message, skip over all others without retransmitting the loop message.
;   434        0431  2   ! This is in case the target is having trouble receiving the loop messae
;   435        0432  2   ! (buffer's too small, for example), and is retransmitting the OC message.
;   436        0433  2   ! If this MOM doesn't wait for them, another MOM gets started up by the
;   437        0434  2   ! OC message, and things start to ping-pong indefinitely.
;   438        0435  2   !
;   439        0436  2   retry = 2;
;   440        0437  2   skip_invalid_MOP_msg = 0;
;   441        0438  2   INCR i FROM 0 TO .loop_count - 1 DO
;   442        0439  3       BEGIN
;   443        0440  3       !
;   444        0441  3       ! Send and receive loop messages retrying on errors.
;   445        0442  3       !
;   446        0443  3       WHILE .retry GTR 0 DO
;   447        0444  4           BEGIN
;   448        0445  4           status = mom$mopsndrcv (.mom_xmit_cib, snddsc,
;   449        0446  4                                   .mom_recv_cib, rcvdsc,
;   450        0447  4                                   rcvlen,
;   451        0448  4                                   .skip_invalid_MOP_msg);
;  452   0449 4         !
;  453   0450 4         ! If a message was received successfully then make sure that it matches
;  454   0451 4         ! what was sent.  If it does, count one message successfully looped.
;  455   0452 4         !
;  456   0453 4         IF .status THEN
;  457   0454 5             BEGIN
;  458   0455 5             IF .rcvlen EQL 1 AND
;  459   0456 5                .(.rcvdsc [1])<0,8> EQL %X'0C' THEN
;  460   0457 5                 skip_invalid_MOP_msg = UPLIT (1, UPLIT BYTE (%X'0C'))
;  461   0458 5             ELSE
;  462   0459 6                 BEGIN
;  463   0460 6                 status = mom$chkbuffer (.snddsc [0],
;  464   0461 6                                          .snddsc [1],
;  465   0462 6                                          .rcvlen,
;  466   0463 6                                          .rcvdsc [1],
;  467   0464 6                                         mom$c_loop_mop);
;  468   0465 6                 IF .status THEN
;  469   0466 6                     EXITLOOP;
;  470   0467 5                 END;
;  471   0468 5             END
;  472   0469 4         ELSE
;  473   0470 4             !
;  474   0471 4             ! If the transmission of the loop message completed with any error
;  475   0472 4             ! except a time out, quit trying right away.
;  476   0473 4             !
;  477   0474 5             BEGIN
;  478   0475 5             IF .status NEQ ss$_timeout THEN
;  479   0476 5                 EXITLOOP;
;  480   0477 4             END;
;  481   0478 4         !
;  482   0479 4         ! If an error was encountered then set up the error response message
;  483   0480 4         ! and keep trying to send this one as long as the retry count allows.
;  484   0481 4         !
;  485   0482 4         retry = .retry - 1;
;  486   0483 4         mom$ab_msgblock [msb$b_code]  = nma$c_sts_lpr;
;  487   0484 3         END;
;  488   0485 3 !
;  489   0486 3 ! If a loop message was not successfully sent and received even after
;  490   0487 3 ! retries then stop trying to loop.
;  491   0488 3 !
;  492   0489 3     IF NOT .status THEN
;  493   0490 4         BEGIN
;  494   0491 4         msgs_looped = .loop_count - .i; ! Set up of count of messages
;  495   0492 4         EXITLOOP;
;  496   0493 3         END;
;  497   0494 2     END;
;  498   0495 2 !
;  499   0496 2 ! If all the loop messages were sent and receive successfully then
;  500   0497 2 ! return a success message.  If errors were encountered, return a
;  501   0498 2 ! message indicating the error and the number of messages not looped.
;  502   0499 2 !
;  503   0500 2 IF .status THEN
;  504   0501 3     BEGIN
;  505   0502 3     mom$ab_msgblock [msb$l_flags] = 0;
;  506   0503 3     mom$ab_msgblock [msb$b_code]  = nma$c_sts_suc;
;  507   0504 3     END
;  508   0505 2 ELSE
;  509      0506  3         BEGIN
;  510      0507  3         mom$ab_msgblock [msb$l_flags] = .mom$ab_msgblock [msb$l_flags] OR
;  511      0508  3                         msb$m_data_fld;
;  512      0509  3         mom$ab_msgblock [msb$a_data]     = UPLIT (2, msgs_looped);
;  513      0510  2         END;
;  514      0511  2   !
;  515      0512  2   ! Build and signal the response message.
;  516      0513  2   !
;  517      0514  2   mom$bld_reply (mom$ab_msgblock, msgsize);
;  518      0515  2   $signal_msg (mom$ab_nice_xmit_buf, .msgsize);
;  519      0516  2
;  520      0517  1   END;                                 ! End of mom$activeloop



                                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2

                                              0C  00058 P.AAH:    .BYTE   12                                         ;
                                                  00059           .BLKB   3
                                        00000001  0005C P.AAG:    .LONG   1                                          ;
                                        00000000' 00060           .ADDRESS P.AAH                                     ;
                                        00000002  00064 P.AAI:    .LONG   2                                          ;
                                        00000000' 00068           .ADDRESS MSGS_LOOPED                               ;

                                                                  .PSECT  $OWN$,NOEXE,2

                                                  001BE           .BLKB   2
                                                  001C0 MSGS_LOOPED:
                                                                  .BLKB   4



                                                                  .PSECT  $CODE$,NOWRT,2

                                            0FFC  00000           .ENTRY  MOM$ACTIVELOOP, Save R2,R3,R4,R5,R6,R7,R8,- ; 0279
                                                                          R9,R10,R11                                  ;
                      5B 00000000G    00    9E  00002             MOVAB   MOM$AB_LOOP_CIB, R11                        ;
                      5A 00000000G    00    9E  00009             MOVAB   MOM$AB_MSGBLOCK, R10                        ;
                      59 00000000G    00    9E  00010             MOVAB   MOM$AB_CIB, R9                              ;
                      5E              18    C2  00017             SUBL2   #24, SP                                     ;
                      6D       0204   CF    DE  0001A             MOVAL   22$, (FP)                                   ; 0297
           00000000G  00              00    FB  0001F             CALLS   #0, MOM$GET_CIRCUIT_TYPE                    ; 0319
           00000000G  00              00    FB  00026             CALLS   #0, MOM$GETSRVTIMER                         ; 0320
        24 00000000G  00              01    E0  0002D             BBS     #1, MOM$GL_SERVICE_FLAGS, 1$                ; 0321
                                      59    DD  00035             PUSHL   R9                                          ; 0328
           00000000G  00              01    FB  00037             CALLS   #1, MOM$MOPOPEN                             ;
                                      7E    7C  0003E             CLRQ    -(SP)                                       ; 0329
                      7E              12    7D  00040             MOVQ    #18, -(SP)                                  ;
                                      59    DD  00043             PUSHL   R9                                          ;
           00000000G  00              05    FB  00045             CALLS   #5, MOM$INIT_CIB                            ;
                  10  A9              01    88  0004C             BISB2   #1, MOM$AB_CIB+16                           ; 0330
                      57              69    9E  00050             MOVAB   MOM$AB_CIB, MOM_RECV_CIB                    ; 0331
                      56              57    D0  00053             MOVL    MOM_RECV_CIB, MOM_XMIT_CIB                  ;
                                    00F4    31  00056             BRW     12$                                         ; 0321
        0E 00000000G  00              03    E1  00059 1$:         BBC     #3, MOM$GL_SERVICE_FLAGS, 2$                ; 0341
                      07 00000000*    00    E8  00061             BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPH*137>-; 0342
                                                                          >+7>, 2$                                    ;
        00000000*  00              02  D0 00068        MOVL    #2, <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPH*-; 0343
                                                               137>>+9>
                52 00000000*  00   D0 0006F 2$:        MOVL    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPH*137>-; 0344
                                                               >+9>, HELP_TYPE
                19 00000000*  00   E8 00076            BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_PHA*137>-; 0351
                                                               >+7>, 3$
                   00000000G  8F   DD 0007D            PUSHL   #SVD$GK_PCNO_$LNH                           ; 0352
                   00000000G  8F   DD 00083            PUSHL   #SVD$GK_PCNO_$LNA                           ;
                   00000000G  8F   DD 00089            PUSHL   #SVD$GK_PCNO_LPN                            ;
        00000000G  00              03  FB 0008F        CALLS   #3, MOM$GET_NODE_ID                         ;
                   02              52  D1 00096 3$:    CMPL    HELP_TYPE, #2                               ; 0355
                                   26  13 00099        BEQL    4$                                          ;
                                   59  DD 0009B        PUSHL   R9                                          ; 0357
        00000000G  00              01  FB 0009D        CALLS   #1, MOM$MOPOPEN                             ;
                   00000000G  8F   DD 000A4            PUSHL   #SVD$GK_PCNO_$LNH                           ; 0358
                   00000000G  8F   DD 000AA            PUSHL   #SVD$GK_PCNO_LPN                            ;
                   00000000G  8F   DD 000B0            PUSHL   #SVD$GK_PCNO_PHA                            ;
                                   12  DD 000B6        PUSHL   #18                                         ;
                                   59  DD 000B8        PUSHL   R9                                          ;
        00000000G  00              05  FB 000BA        CALLS   #5, MOM$INIT_CIB                            ;
        FFFFFFFF   8F              52  D1 000C1 4$:    CMPL    HELP_TYPE, #-1                              ; 0369
                                   4A  13 000C8        BEQL    6$                                          ;
                                   5B  DD 000CA        PUSHL   R11                                         ; 0371
        00000000G  00              01  FB 000CC        CALLS   #1, MOM$MOPOPEN                             ;
                19 00000000*  00   E8 000D3            BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPA*137>-; 0372
                                                               >+7>, 5$
                   00000000G  8F   DD 000DA            PUSHL   #SVD$GK_PCNO_$LAH                           ; 0373
                   00000000G  8F   DD 000E0            PUSHL   #SVD$GK_PCNO_$LNN                           ;
                   00000000G  8F   DD 000E6            PUSHL   #SVD$GK_PCNO_LAN                            ;
        00000000G  00              03  FB 000EC        CALLS   #3, MOM$GET_NODE_ID                         ;
                   00000000G  8F   DD 000F3 5$:        PUSHL   #SVD$GK_PCNO_$LNH                           ; 0376
                   00000000G  8F   DD 000F9            PUSHL   #SVD$GK_PCNO_LAN                            ;
                   00000000G  8F   DD 000FF            PUSHL   #SVD$GK_PCNO_LPA                            ;
                                   12  DD 00105        PUSHL   #18                                         ;
                                   5B  DD 00107        PUSHL   R11                                         ;
        00000000G  00              05  FB 00109        CALLS   #5, MOM$INIT_CIB                            ;
               10  AB              01  88 00110        BISB2   #1, MOM$AB_LOOP_CIB+16                      ; 0387
        FFFFFFFF   8F              52  D1 00114 6$:    CMPL    HELP_TYPE, #-1                              ; 0392
                                   05  13 0011B        BEQL    7$                                          ;
                   01              52  D1 0011D        CMPL    HELP_TYPE, #1                               ;
                                   03  12 00120        BNEQ    8$                                          ;
                   56              69  9E 00122 7$:    MOVAB   MOM$AB_CIB, MOM_XMIT_CIB                    ; 0396
                                   52  D5 00125 8$:    TSTL    HELP_TYPE                                   ; 0397
                                   05  13 00127        BEQL    9$                                          ;
                   02              52  D1 00129        CMPL    HELP_TYPE, #2                               ;
                                   03  12 0012C        BNEQ    10$                                         ;
                   56              6B  9E 0012E 9$:    MOVAB   MOM$AB_LOOP_CIB, MOM_XMIT_CIB               ; 0401
                                   52  D5 00131 10$:   TSTL    HELP_TYPE                                   ; 0402
                                   08  15 00133        BLEQ    11$                                         ;
                   02              52  D1 00135        CMPL    HELP_TYPE, #2                               ;
                                   03  14 00138        BGTR    11$                                         ;
                   57              6B  9E 0013A        MOVAB   MOM$AB_LOOP_CIB, MOM_RECV_CIB               ; 0406
        FFFFFFFF   8F              52  D1 0013D 11$:   CMPL    HELP_TYPE, #-1                              ; 0407
                                   07  19 00144        BLSS    12$                                         ;
                                   52  D5 00146        TSTL    HELP_TYPE                                   ;
                                   03  14 00148        BGTR    12$                                         ;
                   57              69  9E 0014A        MOVAB   MOM$AB_CIB, MOM_RECV_CIB                    ; 0411
```

```
                                         66  DD 0014D 12$:   PUSHL    (MOM_XMIT_CIB)                                ; 0418
                                         02  DD 0014F        PUSHL    #2                                            ; 0417
                  00000000G  00          02  FB 00151        CALLS    #2, MOM$MOPSETSUBSTATE                        ;
                                  08     AE  9F 00158        PUSHAB   RCVDSC                                        ; 0419
                                  14     AE  9F 0015B        PUSHAB   SNDDSC                                        ;
                  00000000V  00          02  FB 0015E        CALLS    #2, MOM_BUILD_LOOP_BUFS                       ;
                             55 00000000*  00  B0 00165      MOVW     <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPC*137>-; 0424
                                                                      >+9>, LOOP_COUNT                              ;
                             54          02  D0 0016C        MOVL     #2, RETRY                                     ; 0436
                                         53  D4 0016F        CLRL     SKIP_INVALID_MOP_MSG                          ; 0437
                             58          55  3C 00171        MOVZWL   LOOP_COUNT, R8                                ; 0438
                             52          01  CE 00174        MNEGL    #1, I                                         ; 0446
                                         6C  11 00177        BRB      18$                                           ;
                                         54  D5 00179 13$:   TSTL     RETRY                                         ; 0443
                                         58  15 0017B        BLEQ     17$                                           ;
                                         53  DD 0017D        PUSHL    SKIP_INVALID_MOP_MSG                          ; 0448
                                  04     AE  9F 0017F        PUSHAB   RCVLEN                                        ; 0445
                                  10     AE  9F 00182        PUSHAB   RCVDSC                                        ;
                                         57  DD 00185        PUSHL    MOM_RECV_CIB                                  ; 0446
                                  20     AE  9F 00187        PUSHAB   SNDDSC                                        ; 0445
                                         56  DD 0018A        PUSHL    MOM_XMIT_CIB                                  ;
                  00000000G  00          06  FB 0018C        CALLS    #6, MOM$MOPSNDRCV                             ;
                             2E          50  E9 00193        BLBC     STATUS, 15$                                   ; 0453
                             01          6E  D1 00196        CMPL     RCVLEN, #1                                    ; 0455
                                         0F  12 00199        BNEQ     14$                                           ;
                             0C   0C     BE  91 0019B        CMPB     @RCVDSC+4, #12                                ; 0456
                                         09  12 0019F        BNEQ     14$                                           ;
                             53 00000000'  00  9E 001A1      MOVAB    P.AAG, SKIP_INVALID_MOP_MSG                   ; 0457
                                         23  11 001A8        BRB      16$                                           ;
                                         7E  D4 001AA 14$:   CLRL     -(SP)                                         ; 0460
                                  10     AE  DD 001AC        PUSHL    RCVDSC+4                                      ; 0463
                                  08     AE  DD 001AF        PUSHL    RCVLEN                                        ; 0462
                                  20     AE  DD 001B2        PUSHL    SNDDSC+4                                      ; 0461
                                  20     AE  DD 001B5        PUSHL    SNDDSC                                        ; 0460
                  00000000V  00          05  FB 001B8        CALLS    #5, MOM$CHKBUFFER                             ;
                             0B          50  E9 001BF        BLBC     STATUS, 16$                                   ; 0465
                                         11  11 001C2        BRB      17$                                           ; 0466
                  0000022C   8F          50  D1 001C4 15$:   CMPL     STATUS, #556                                  ; 0475
                                         08  12 001CB        BNEQ     17$                                           ;
                                         54  D7 001CD 16$:   DECL     RETRY                                         ; 0482
                         04  AA          11  8E 001CF        MNEGB    #17, MOM$AB_MSGBLOCK+4                        ; 0483
                                         A4  11 001D3        BRB      13$                                           ; 0443
                             0D          50  E8 001D5 17$:   BLBS     STATUS, 18$                                   ; 0489
                             51          55  3C 001D8        MOVZWL   LOOP_COUNT, R1                                ; 0491
       00000000'  00         51          52  C3 001DB        SUBL3    I, R1, MSGS_LOOPED                            ;
                                         04  11 001E3        BRB      19$                                           ; 0490
                  90         52          58  F2 001E5 18$:   AOBLSS   R8, I, 13$                                    ; 0438
                             08          50  E9 001E9 19$:   BLBC     STATUS, 20$                                   ; 0500
                                         6A  D4 001EC        CLRL     MOM$AB_MSGBLOCK                               ; 0502
                         04  AA          01  90 001EE        MOVB     #1, MOM$AB_MSGBLOCK+4                         ; 0503
                                         0B  11 001F2        BRB      21$                                           ; 0500
                             6A          20  88 001F4 20$:   BISB2    #32, MOM$AB_MSGBLOCK                          ; 0507
                         18  AA 00000000'  00  9E 001F7      MOVAB    P.AAI, MOM$AB_MSGBLOCK+24                     ; 0509
                                  04     AE  9F 001FF 21$:   PUSHAB   MSGSIZE                                       ; 0514
                                         5A  DD 00202        PUSHL    R10                                           ;
                  00000000G  00          02  FB 00204        CALLS    #2, MOM$BLD_REPLY                             ;
                                  04     AE  DD 0020B        PUSHL    MSGSIZE                                       ; 0515
```

```
                          00000000G  00  9F 0020E          PUSHAB  MOM$AB_NICE_XMIT_BUF             ;
                          02070000   8F  DD 00214          PUSHL   #34013184                        ;
                00000000G  00            03  FB 0021A          CALLS   #3, LIB$SIGNAL                   ;
                                             04 00221          RET                                      ; 0517
                                           0000 00222 22$:     .WORD   Save nothing                     ; 0297
                                         7E  D4 00224          CLRL    -(SP)                            ;
                                         5E  DD 00226          PUSHL   SP                               ;
                           7E        04  AC  7D 00228          MOVQ    4(AP), -(SP)                     ;
                00000000V  00            03  FB 0022C          CALLS   #3, MOM$LOOPHANDLER              ;
                                             04 00233          RET                                      ;

; Routine Size:  564 bytes,    Routine Base:  $CODE$ + 0069
;  522        0518   1 %SBTTL 'mom_build_loop_bufs    Build buffer containing loop message'
;  523        0519   1 GLOBAL ROUTINE mom_build_loop_bufs (xmit_buffer_dsc,
;  524        0520   1                                     recv_buffer_dsc) : NOVALUE =
;  525        0521   1
;  526        0522   1 !++
;  527        0523   1 ! FUNCTIONAL DESCRIPTION:
;  528        0524   1 !
;  529        0525   1 !       This routine is called to build a circuit loop message.  It gets a
;  530        0526   1 !       buffer of the appropriate size and fills in the loop message.
;  531        0527   1 !
;  532        0528   1 ! IMPLICIT INPUTS:
;  533        0529   1 !       xmit_buffer_dsc - Address at which to return descriptor of completed
;  534        0530   1 !               loop message to transmit.
;  535        0531   1 !       recv_buffer_dsc - Address at which to return descriptor of completed
;  536        0532   1 !               buffer.
;  537        0533   1 !
;  538        0534   1 ! ROUTINE VALUE:
;  539        0535   1 ! COMPLETION CODES:
;  540        0536   1 !       Signal errors.
;  541        0537   1 !
;  542        0538   1 !--
;  543        0539   1
;  544        0540   2 BEGIN
;  545        0541   2
;  546        0542   2 OWN
;  547        0543   2     loop_size;
;  548        0544   2
;  549        0545   2 LOCAL
;  550        0546   2     loop_data   : BYTE,
;  551        0547   2     msgsize;
;  552        0548   2
;  553        0549   2 !
;  554        0550   2 ! Request buffers for loopback.
;  555        0551   2 !
;  556        0552   2 loop_size = .mom$ab_service_data [svd$gk_pcno_lpl, svd$l_param];
;  557        0553   2 !
;  558        0554   2 ! If the circuit is an NI, the loop size doesn't include the headers.
;  559        0555   2 ! Add the maximum possible header size to the loop size.
;  560        0556   2 !
;  561        0557   2 IF .mom$gl_service_flags [mom$v_ni_circ] THEN
;  562        0558   2     loop_size = .loop_size + mom$k_max_loop_header;
;  563        0559   2
;  564        0560   2 IF NOT mom$getbuffer (loop_size, .xmit_buffer_dsc, .recv_buffer_dsc) THEN
;  565        0561   3     BEGIN
;  566        0562   3     mom$ab_msgblock [msb$l_flags]  = msb$m_det_fld or
;  567        0563   3                                      msb$m_msg_fld or
;  568        0564   3                                      msb$m_data_fld;
;  569        0565   3     mom$ab_msgblock [msb$b_code]   = nma$c_sts_pva;
;  570        0566   3     mom$ab_msgblock [msb$w_detail] = nma$c_pcno_lpl;
;  571        0567   3     mom$ab_msgblock [msb$l_text]   = mom$_alpbfovf;
;  572        0568   3     mom$ab_msgblock [msb$a_data]   = UPLIT (2, loop_size);
;  573        0569   3
;  574        0570   3     mom$bld_reply (mom$ab_msgblock, msgsize);
;  575        0571   3     $signal_msg (mom$ab_nice_xmit_buf, .msgsize);
;  576        0572   2     END;
;  577        0573   2 !
;  578        0574   2 ! Initialize the transmit buffer with the specified data.
;  579        0575  2 !
;  580        0576  2 loop_data = .mom$ab_service_data [svd$gk_pcno_lpd, svd$l_param];
;  581        0577  2
;  582        0578  2 mom$initbuffer (.xmit_buffer_dsc, .loop_data, mom$c_loop_mop);
;  583        0579  1 END;                                   ! End of mom_build_loop_bufs


                                                                   .PSECT  $PLIT$,NOWRT,NOEXE,2

                                        00000002  0006C P.AAJ:    .LONG   2                                                  ;
                                        00000000' 00070           .ADDRESS LOOP_SIZE                                         ;

                                                                   .PSECT  $OWN$,NOEXE,2

                                                  001C4 LOOP_SIZE:
                                                                   .BLKB   4


                                                                   .PSECT  $CODE$,NOWRT,2

                                             000C 00000            .ENTRY  MOM_BUILD_LOOP_BUFS, Save R2,R3                   ; 0519
                          53 00000000'  00    9E 00002            MOVAB   LOOP_SIZE, R3                                      ;
                          52 00000000G  00    9E 00009            MOVAB   MOM$AB_MSGBLOCK, R2                                ;
                          5E            04    C2 00010            SUBL2   #4, SP                                             ;
                          63 00000000*  00    D0 00013            MOVL    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPL*137>-; 0552
                                                                          >+9>, LOOP_SIZE                                    ;
           03 00000000G   00            01    E1 0001A            BBC     #1, MOM$GL_SERVICE_FLAGS, 1$                       ; 0557
                          63            1E    C0 00022            ADDL2   #30, LOOP_SIZE                                     ; 0558
                          7E      04    AC    7D 00025 1$:        MOVQ    XMIT_BUFFER_DSC, -(SP)                             ; 0560
                                        53    DD 00029            PUSHL   R3                                                 ;
              00000000V   00            03    FB 0002B            CALLS   #3, MOM$GETBUFFER                                  ;
                          3C            50    E8 00032            BLBS    R0, 2$                                             ;
                          62            26    D0 00035            MOVL    #38, MOM$AB_MSGBLOCK                               ; 0563
                    04    A2            10    8E 00038            MNEGB   #16, MOM$AB_MSGBLOCK+4                             ; 0565
                    08    A2      97    8F    9B 0003C            MOVZBW  #151, MOM$AB_MSGBLOCK+8                            ; 0566
                    0C    A2 00000000G  8F    D0 00041            MOVL    #MOM$_ALPBFOVF, MOM$AB_MSGBLOCK+12                 ; 0567
                    18    A2 00000000'  00    9E 00049            MOVAB   P.AAJ, MOM$AB_MSGBLOCK+24                          ; 0568
                                  4004  8F    BB 00051            PUSHR   #^M<R2,SP>                                         ; 0570
              00000000G   00            02    FB 00055            CALLS   #2, MOM$BLD_REPLY                                  ;
                                        6E    DD 0005C            PUSHL   MSGSIZE                                            ; 0571
                             00000000G  00    9F 0005E            PUSHAB  MOM$AB_NICE_XMIT_BUF                               ;
                             02070000   8F    DD 00064            PUSHL   #34013184                                          ;
              00000000G   00            03    FB 0006A            CALLS   #3, LIB$SIGNAL                                     ;
                          50 00000000*  00    90 00071 2$:        MOVB    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPD*137>-; 0576
                                                                          >+9>, LOOP_DATA                                    ;
                                        7E    D4 00078            CLRL    -(SP)                                              ; 0578
                          7E            50    9A 0007A            MOVZBL  LOOP_DATA, -(SP)                                   ;
                                  04    AC    DD 0007D            PUSHL   XMIT_BUFFER_DSC                                    ;
              00000000V   00            03    FB 00080            CALLS   #3, MOM$INITBUFFER                                 ;
                                              04 00087            RET                                                        ; 0579

; Routine Size:  136 bytes,     Routine Base:  $CODE$ + 029D
```

```
1 %SBTTL 'mom$passiveloop  Passive loop'
1 GLOBAL ROUTINE mom$passiveloop (mop_msg_dsc) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine performs the passive loopback function.
1 !       It provides the mirror function for the LOOP CIRCUIT command.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       Signal errors.
1 !
1 !--
1
2 BEGIN
2
2 MAP
2     mop_msg_dsc : ref vector;
2
2 LOCAL
2     status;
2
2 !
2 ! Enable condition handler to clean up after loop operation.
2 !
2 ENABLE mom$loophandler;
2 !
2 ! Log the event indicating that a passive loopback operation was initiated.
2 !
2 mom$gw_evt_code = evc$c_nma_psl;        ! Event code (passive loopback)
2 mom$gb_evt_popr = evc$c_nma_popr_ini;! Operation (initiated)
2 mom$log_event (0, 0);
2 mom$gb_evt_popr = evc$c_nma_popr_ter;! Operation (terminated)
2 !
2 ! Set the circuit substate to -REFLECTING.
2 !
2 mom$mopsetsubstate (nma$c_linss_ref,
2                     .mom$ab_cib [cib$l_chan]);
2 mom$ab_cib [cib$l_retry_cnt] = 1;
2 WHILE T DO
3     BEGIN
3     CH$WCHAR (mop$_fct_pld, .mop_msg_dsc [1]);
3
3     status = mom$mopsndrcv (mom$ab_cib, .mop_msg_dsc,
3                             mom$ab_cib, mom$gq_mop_rcv_buf_dsc,
3                             mop_msg_dsc [0],
3                             0);                 ! Don't skip any received msgs.
3     mom$chk_mop_error (.status);
3
3     IF .(.mop_msg_dsc [1])<0,8> NEQU mop$_fct_ald THEN
3         EXITLOOP;
3
2     END;
2
2 mom$error (nma$c_sts_suc);
```

```
;  642       0637  2
;  643       0638  1 END;                                   ! End of mom$passiveloop




                                          003C 00000            .ENTRY   MOM$PASSIVELOOP, Save R2,R3,R4,R5             ; 0581
                   55 00000000G      00    9E 00002            MOVAB    MOM$GB_EVT_POPR, R5                           ;
                   54 00000000G      00    9E 00009            MOVAB    MOM$AB_CIB, R4                                ;
                   6D       0062     CF    DE 00010            MOVAL    2$, (FP)                                      ; 0596
          00000000G  00              06    90 00015            MOVB     #6, MOM$GW_EVT_CODE                           ; 0611
                     65              94 0001C                  CLRB     MOM$GB_EVT_POPR                               ; 0612
                                     7E    7C 0001E            CLRQ     -(SP)                                         ; 0613
          00000000G  00              02    FB 00020            CALLS    #2, MOM$LOG_EVENT                             ;
                     65              01    90 00027            MOVB     #1, MOM$GB_EVT_POPR                           ; 0614
                                     64    DD 0002A            PUSHL    MOM$AB_CIB                                    ; 0619
                                     01    DD 0002C            PUSHL    #1                                            ; 0618
          00000000G  00              02    FB 0002E            CALLS    #2, MOM$MOPSETSUBSTATE                        ;
                 12  A4              01    D0 00035            MOVL     #1, MOM$AB_CIB+18                             ; 0620
                     52        04    AC    D0 00039            MOVL     MOP_MSG_DSC, R2                               ; 0623
                 04  B2              19    90 0003D 1$:        MOVB     #25, @4(R2)                                   ;
                                     7E    D4 00041            CLRL     -(SP)                                         ; 0627
                               04    AC    DD 00043            PUSHL    MOP_MSG_DSC                                   ;
                        00000000G    00    9F 00046            PUSHAB   MOM$GQ_MOP_RCV_BUF_DSC                        ; 0625
                                     54    DD 0004C            PUSHL    R4                                            ;
                               04    AC    DD 0004E            PUSHL    MOP_MSG_DSC                                   ; 0627
                                     54    DD 00051            PUSHL    R4                                            ; 0625
          00000000G  00              06    FB 00053            CALLS    #6, MOM$MOPSNDRCV                             ; 0627
                     53              50    D0 0005A            MOVL     R0, STATUS                                    ;
                                     53    DD 0005D            PUSHL    STATUS                                        ; 0629
          00000000G  00              01    FB 0005F            CALLS    #1, MOM$CHK_MOP_ERROR                         ;
                     18        04    B2    91 00066            CMPB     @4(R2), #24                                   ; 0631
                                     D1    13 0006A            BEQL     1$                                            ;
                                     01    DD 0006C            PUSHL    #1                                            ; 0636
          00000000G  00              01    FB 0006E            CALLS    #1, MOM$ERROR                                 ;
                                          04 00075            RET                                                    ; 0638
                                        0000 00076 2$:        .WORD    Save nothing                                  ; 0596
                                     7E    D4 00078            CLRL     -(SP)                                         ;
                                     5E    DD 0007A            PUSHL    SP                                            ;
                     7E        04    AC    7D 0007C            MOVQ     4(AP), -(SP)                                  ;
          00000000V  00              03    FB 00080            CALLS    #3, MOM$LOOPHANDLER                           ;
                                          04 00087            RET                                                    ;

; Routine Size:  136 bytes,    Routine Base:  $CODE$ + 0325
```

```
      1 %SBTTL 'mom$loop_line  Loop PSI Line'
      1 GLOBAL ROUTINE mom$loop_line : NOVALUE =
      1
      1 !++
      1 ! FUNCTIONAL DESCRIPTION:
      1 !       This routine is called when MOM receives a LOOP LINE command from NCP.
      1 !       Loop line can only be performed on PSI (LAPB) lines.  This routine
      1 !       builds buffers and issues a QIOW to tell the PSIACP to perform the
      1 !       loop line function.  It then takes the completion status of the QIO
      1 !       and builds and sends a NICE response message to NCP.
      1 !
      1 ! IMPLICIT INPUTS:
      1 !       MOM$GQ_ENTITY_BUF_DSC   Contains descriptor of loop line ID.
      1 !
      1 ! ROUTINE VALUE:
      1 ! COMPLETION CODES:
      1 !       Signal errors.
      1 !
      1 !--
      1
      2 BEGIN
      2
P     2 $nfbdsc (mom_q_loop_line_nfb, loop, , pli
P     2         ,nam,           ! Search key one = line name, oper1 = eql
P     2         ,,              ! Null search key two
P     2
P     2         ,lpc            ! Loop count
P     2         ,lpl            ! Loop message length
P     2         ,lpd            ! Loop message data type
      2         );
      2 MAP
      2     mom_q_loop_line_nfb : VECTOR;
      2
      2 LOCAL
      2     status,
      2     p2_dsc:         VECTOR [2],
      2     p4_buf_dsc: VECTOR [2],
      2     p4_buffer:  VECTOR [3],
      2     p3,
      2     iosb:           $iosb,
      2     database,
      2     loop_line_chan: WORD,
      2     nfb :           REF BBLOCK,
      2     msgsize;
      2
      2 !
      2 ! Build the loop line qio buffers.
      2 !
      2 mom$build_p2 (
      2         .mom$gq_entity_buf_dsc [0],                     ! Search key one length
      2         .mom$gq_entity_buf_dsc [1],                     ! Search key one address
      2         -1, 0,
      2         mom$q_p2bfdsc, p2_dsc);
      2
      2 p4_buffer [0] = .mom$ab_service_data [svd$gk_pcno_lpc, svd$l_param];
      2 p4_buffer [1] = .mom$ab_service_data [svd$gk_pcno_lpl, svd$l_param];
      2 p4_buffer [2] = .mom$ab_service_data [svd$gk_pcno_lpd, svd$l_param];
```

```
;  702      0696  2 p4_buf_dsc [0] = 12;
;  703      0697  2 p4_buf_dsc [1] = p4_buffer;
;  704      0698  2
;  705      0699  2 !
;  706      0700  2 ! Assign a channel directly to PSI rather than getting to PSI via NETACP,
;  707      0701  2 ! which is the normal route used for SET, SHOW etc QIOs issued by NML.
;  708      0702  2 ! In this case this isn't appropriate because NETACP can't afford to wait
;  709      0703  2 ! around for a loop function complete (or not, whichever the case may be).
;  710      0704  2 ! So, for loop line, MOM bypasses NETACP.
;  711      0705  2 !
;  712    P 0706  2 status = $ASSIGN (CHAN = loop_line_chan,
;  713      0707  2                   DEVNAM = mom$gq_psinamdsc);
;  714      0708  2 IF NOT .status THEN
;  715      0709  3     BEGIN
;  716      0710  3     mom$ab_msgblock [msb$l_flags] = msb$m_msg_fld;
;  717      0711  3     mom$ab_msgblock [msb$b_code] = nma$c_sts_lco;
;  718      0712  3     mom$ab_msgblock [msb$l_text] = .status;
;  719      0713  3     END
;  720      0714  2 ELSE
;  721      0715  3     BEGIN
;  722      0716  3     !
;  723      0717  3     ! Issue the loop line QIOW to PSIACP.
;  724      0718  3     !
;  725    P 0719  3     status = $QIOW (CHAN = .loop_line_chan,
;  726    P 0720  3                     FUNC = io$_acpcontrol,
;  727    P 0721  3                     IOSB = iosb,
;  728    P 0722  3                     P1 = mom_q_loop_line_nfb,
;  729    P 0723  3                     P2 = p2_dsc,
;  730    P 0724  3                     P3 = p3,
;  731      0725  3                     P4 = p4_buf_dsc);
;  732      0726  3     !
;  733      0727  3     ! If logging is turned on, dump the QIO status and buffers.
;  734      0728  3     !
;  735      0729  3     mom$debug_qio (dbg$c_acpqio,                    ! Log type code
;  736      0730  3                 .status,                          ! QIO completion status
;  737      0731  3                 iosb,                             ! I/O status block
;  738      0732  3                 mom_q_loop_line_nfb,              ! NFB buffer descriptor
;  739      0733  3                 p2_dsc,                           ! P2 buffer descriptor
;  740      0734  3                 p3,                               ! length of data in P4 buffer
;  741      0735  3                 p4_buf_dsc);                      ! P4 buffer descriptor
;  742      0736  3
;  743      0737  3     !
;  744      0738  3     ! Using the QIO completion status, set up the return status for
;  745      0739  3     ! the NICE response message to NCP.
;  746      0740  3     !
;  747      0741  3     nfb = .mom_q_loop_line_nfb [1];
;  748      0742  3     database = .nfb [nfb$b_database];
;  749      0743  3     IF mom$mapqioerror (.database, .status, iosb) THEN
;  750      0744  4         BEGIN
;  751      0745  4         mom$ab_msgblock [msb$l_flags] = 0;
;  752      0746  4         mom$ab_msgblock [msb$b_code] = nma$c_sts_suc;
;  753      0747  3         END;
;  754      0748  3     !
;  755      0749  3     ! Deassign the QIO channel to NW:
;  756      0750  3     !
;  757      0751  3     $DASSGN (CHAN = .loop_line_chan);
;  758      0752  2     END;
;  759            0753  2 !
;  760            0754  2 ! Build and signal the NICE response message to NCP.
;  761            0755  2 !
;  762            0756  2 mom$bld_reply (mom$ab_msgblock, msgsize);
;  763            0757  2 $signal_msg (mom$ab_nice_xmit_buf, .msgsize);
;  764            0758  1 END;                                  ! End of MOM$LOOPLINE


                                                                                   .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                       00000024  00074 P.AAK:      .LONG   36                                            ;
                                                       00000000' 00078             .ADDRESS U.1                                          ;

                                                                                   .PSECT  $OWN$,NOEXE,2

                                                             26  001C8 ;_NFB
                                                                       U.1:        .BYTE   38                                            ;
                                                             00  001C9             .BYTE   0                                             ;
                                                             05  001CA             .BYTE   5                                             ;
                                                             00  001CB             .BYTE   0                                             ;
                                                       05020041  001CC             .LONG   84017217                                      ;
                                                       00000001  001D0             .LONG   1                                             ;
                                                             00  001D4             .BYTE   0                                             ;
                                                             00  001D5             .BYTE   0                                             ;
                                                           0000  001D6             .WORD   0                                             ;
                                                       05010023  001D8             .LONG   83951651                                      ;
                                                       05010024  001DC             .LONG   83951652                                      ;
                                                       05010025  001E0             .LONG   83951653                                      ;
                                                       00000000  001E4             .LONG   0                                             ;
                                                                 001E8             .BLKB   4

                                                                       U.2=                 P.AAK
                                                                                   .EXTRN  SYS$ASSIGN, SYS$QIOW
                                                                                   .EXTRN  SYS$DASSGN

                                                                                   .PSECT  $CODE$,NOWRT,2

                                                           001C  00000             .ENTRY  MOM$LOOP_LINE, Save R2,R3,R4                  ; 0640
                                    54 00000000'     00    9E  00002             MOVAB   MOM_Q_LOOP_LINE_NFB, R4                       ;
                                    53 00000000G     00    9E  00009             MOVAB   MOM$AB_MSGBLOCK, R3                           ;
                                    5E               30    C2  00010             SUBL2   #48, SP                                       ;
                                               28    AE    9F  00013             PUSHAB  P2_DSC                                        ; 0687
                                               94    A4    9F  00016             PUSHAB  MOM$Q_P2BFDSC                                 ;
                                                     7E    D4  00019             CLRL    -(SP)                                         ;
                                    7E               01    CE  0001B             MNEGL   #1, -(SP)                                     ; 0690
                                    7E 00000000G     00    7D  0001E             MOVQ    MOM$GQ_ENTITY_BUF_DSC, -(SP)                  ; 0688
                          00000000G 00               06    FB  00025             CALLS   #6, MOM$BUILD_P2                              ;
                                 14 AE 00000000*     00    D0  0002C             MOVL    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPC*137>-; 0693
                                                                                         >+9>, P4_BUFFER                               ;
                                 18 AE 00000000*     00    D0  00034             MOVL    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPL*137>-; 0694
                                                                                         >+9>, P4_BUFFER+4                             ;
                                 1C AE 00000000*     00    D0  0003C             MOVL    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPD*137>-; 0695
                                                                                         >+9>, P4_BUFFER+8                             ;
                                 20 AE               0C    D0  00044             MOVL    #12, P4_BUF_DSC                               ; 0696
                                 24 AE         14    AE    9E  00048             MOVAB   P4_BUFFER, P4_BUF_DSC+4                       ; 0697
                                                     7E    7C  0004D             CLRQ    -(SP)                                         ; 0707
```

```
                       08  AE 9F 0004F        PUSHAB  LOOP_LINE_CHAN                ;
                 00000000G 00 9F 00052        PUSHAB  MOM$GQ_PSINAMDSC              ;
00000000G  00              04 FB 00058        CALLS   #4, SYS$ASSIGN                ;
           52              50 D0 0005F        MOVL    R0, STATUS                    ;
           0D              52 E8 00062        BLBS    STATUS, 1$                    ; 0708
           63              04 D0 00065        MOVL    #4, MOM$AB_MSGBLOCK           ; 0710
      04   A3              0A 8E 00068        MNEGB   #10, MOM$AB_MSGBLOCK+4        ; 0711
      0C   A3              52 D0 0006C        MOVL    STATUS, MOM$AB_MSGBLOCK+12    ; 0712
                           64 11 00070        BRB     3$                            ; 0708
                           7E 7C 00072 1$:    CLRQ    -(SP)                         ; 0725
                       28  AE 9F 00074        PUSHAB  P4_BUF_DSC                    ;
                       10  AE 9F 00077        PUSHAB  P3                            ;
                       38  AE 9F 0007A        PUSHAB  P2_DSC                        ;
                           54 DD 0007D        PUSHL   R4                            ;
                           7E 7C 0007F        CLRQ    -(SP)                         ;
                       2C  AE 9F 00081        PUSHAB  IOSB                          ;
                           38 DD 00084        PUSHL   #56                           ;
           7E          28  AE 3C 00086        MOVZWL  LOOP_LINE_CHAN, -(SP)         ;
                           7E D4 0008A        CLRL    -(SP)                         ;
00000000G  00              0C FB 0008C        CALLS   #12, SYS$QIOW                 ;
           52              50 D0 00093        MOVL    R0, STATUS                    ;
                       20  AE 9F 00096        PUSHAB  P4_BUF_DSC                    ; 0729
                       08  AE 9F 00099        PUSHAB  P3                            ;
                       30  AE 9F 0009C        PUSHAB  P2_DSC                        ;
                           54 DD 0009F        PUSHL   R4                            ;
                       1C  AE 9F 000A1        PUSHAB  IOSB                          ;
                           52 DD 000A4        PUSHL   STATUS                        ; 0730
                           04 DD 000A6        PUSHL   #4                            ; 0729
00000000G  00              07 FB 000A8        CALLS   #7, MOM$DEBUG_QIO             ;
           50          04  A4 D0 000AF        MOVL    MOM_Q_LOOP_LINE_NFB+4, NFB    ; 0741
           50          02  A0 9A 000B3        MOVZBL  2(NFB), DATABASE              ; 0742
                       0C  AE 9F 000B7        PUSHAB  IOSB                          ; 0743
                           05 BB 000BA        PUSHR   #^M<R0,R2>                    ;
00000000G  00              03 FB 000BC        CALLS   #3, MOM$MAPQIOERROR           ;
           06              50 E9 000C3        BLBC    R0, 2$                        ;
                           63 D4 000C6        CLRL    MOM$AB_MSGBLOCK               ; 0745
      04   A3              01 90 000C8        MOVB    #1, MOM$AB_MSGBLOCK+4         ; 0746
           7E              6E 3C 000CC 2$:    MOVZWL  LOOP_LINE_CHAN, -(SP)         ; 0751
00000000G  00              01 FB 000CF        CALLS   #1, SYS$DASSGN                ;
                       08  AE 9F 000D6 3$:    PUSHAB  MSGSIZE                       ; 0756
                           53 DD 000D9        PUSHL   R3                            ;
00000000G  00              02 FB 000DB        CALLS   #2, MOM$BLD_REPLY             ;
                       08  AE DD 000E2        PUSHL   MSGSIZE                       ; 0757
                 00000000G 00 9F 000E5        PUSHAB  MOM$AB_NICE_XMIT_BUF          ;
                 02070000  8F DD 000EB        PUSHL   #34013184                     ;
00000000G  00              03 FB 000F1        CALLS   #3, LIB$SIGNAL                ;
                              04 000F8        RET                                   ; 0758

; Routine Size:  249 bytes,    Routine Base:  $CODE$ + 03AD
```

```
  1 %SBTTL 'mom$nodetest  Node loopback test'
  1 GLOBAL ROUTINE mom$nodetest : NOVALUE =
  1
  1 !++
  1 ! FUNCTIONAL DESCRIPTION:
  1 !
  1 !       This routine opens a link to the loopback mirror and
  1 !       verifies the connect data to determine the maximum test message size.
  1 !
  1 !--
  1
  2 BEGIN
  2 !
  2 ! Enable the condition handler to perform cleanup.
  2 !
  2 ENABLE mom$testhandler;
  2 !
  2 ! Build the NFB for the QIO IO$_ACPCONTROL
  2 !
P 2 $nfbdsc (nfbdsc,show,,ndi
P 2         ,nna,              ! Search key 1 = node name, oper1 = eql
P 2         ,,                 ! Search key2 = wildcard, oper2 = eql
  2         ,dty);
  2
  2 MAP
  2     nfbdsc          : VECTOR;
  2 OWN
  2     bufsiz,                                 ! Buffer size
  2     msgnotlooped : WORD;                    ! Number of messages not looped
  2
  2 LOCAL
  2     rcv_size,
  2     xmit_iosb  : $IOSB,
  2     rcv_iosb   : $IOSB,
  2     ptr,
  2     rcvdsc     : VECTOR [2],
  2     snddsc     : VECTOR [2],
  2     status,
  2     p2nambuf   : VECTOR [mom$k_p2_buf_len,BYTE], ! Node name buffer for P2
  2                                              !          QIO argument
  2     p2dsc      : VECTOR [2],            ! Descriptor P2 buffer
  2     nfb        : REF BBLOCK,
  2     ent_len,
  2     ent_addr;
  2
  2 !
  2 ! Restructure node name for P2 argument in QIO.
  2 !
  2 nfb = .nfbdsc [1];
  2 IF .mom$gb_entity_code EQL mom$c_nodebyname THEN
  3     BEGIN
  3     ent_len = .mom$gq_entity_buf_dsc [0];
  3     ent_addr = .mom$gq_entity_buf_dsc [1];
  3     nfb [nfb$l_srch_key] = nfb$c_ndi_nna;
  3     END
  2 ELSE
  3     BEGIN
```

```
;  823          0816  3          ent_len = -2;
;  824          0817  3          ent_addr = ..mom$gq_entity_buf_dsc [1];
;  825          0818  3          nfb [nfb$l_srch_key] = nfb$c_ndi_add;
;  826          0819  2          END;
;  827          0820  2     p2dsc [0] = mom$k_p2_buf_len;
;  828          0821  2     p2dsc [1] = p2nambuf;
;  829          0822  2     mom$build_p2 ( .ent_len,                        ! Length of node id
;  830          0823  2                    .ent_addr,                       ! Pointer to node id
;  831          0824  2                    -1, 0,
;  832          0825  2                    p2dsc,                           ! Descriptor of buffer
;  833          0826  2                    p2dsc);                          ! Descriptor of P2 parameter
;  834          0827  2
;  835          0828  2     !
;  836          0829  2     ! Determine which version of mirror will be used.
;  837          0830  2     !
;  838          0831  2     status = mom$netacp_qio
;  839          0832  2                    (nfbdsc,                         ! NFB for QIO IO$_ACPCONTROL
;  840          0833  2                    p2dsc,                           ! Descriptor P2 parameter
;  841          0834  2                    0,
;  842          0835  2                    mom$gq_acpqio_buf_dsc); ! The descriptor pointing to buffer
;  843          0836  2                                            ! where information is returned
;  844          0837  2     IF .status
;  845          0838  2         AND (.mom$ab_acpqio_buffer EQLU nma$c_nodty_pha) THEN
;  846          0839  3         BEGIN
;  847          0840  3         version = mom$c_loop_phase2;         ! If both conditions then Phase 2
;  848          0841  3         bufsiz  = .mom$ab_service_data [svd$gk_pcno_lpl, svd$l_param] + 1;
;  849          0842  3         END
;  850          0843  2     ELSE
;  851          0844  3         BEGIN
;  852          0845  3         version = mom$c_loop_phase3;         ! Else Phase 3 or later
;  853          0846  3         bufsiz  = .mom$ab_service_data [svd$gk_pcno_lpl, svd$l_param];
;  854          0847  2         END;
;  855          0848  2     !
;  856          0849  2     ! Get buffers.
;  857          0850  2     !
;  858          0851  2     status = mom$getbuffer (bufsiz, snddsc, rcvdsc); ! Attempt to allocate it
;  859          0852  2
;  860          0853  2     IF NOT .status THEN
;  861          0854  3         BEGIN
;  862          0855  3         mom_siglooperr (nma$c_sts_pva,
;  863          0856  3                         nma$c_pcno_lpl,
;  864          0857  3                         uplit (2, bufsiz),
;  865          0858  3                         mom$_alpbfovf,
;  866          0859  3                         msb$m_data_fld OR msb$m_msg_fld);
;  867          0860  2         END;
;  868          0861  2     !
;  869          0862  2     ! Build the NFB from the data provided in the command message.
;  870          0863  2     !
;  871          0864  2     mom_bldloopnfb ();
;  872          0865  2     !
;  873          0866  2     ! Attempt to connect to the mirror.
;  874          0867  2     !
;  875          0868  2     mom_openlink ();
;  876          0869  2     !
;  877          0870  2     ! Initialize the transmit buffer according to the loop data type in
;  878          0871  2     ! MOM$AB_SERVICE_DATA [SVD$GK_PCNO_LPD, SVD$L_PARAM].
;  879          0872  2     !
;   880          0873  2 mom$initbuffer (snddsc,
;   881          0874  2                 .mom$ab_service_data [svd$gk_pcno_lpd, svd$l_param],
;   882          0875  2                 .version);
;   883          0876  2 !
;   884          0877  2 ! Start looping the data
;   885          0878  2 !
;   886          0879  2 INCR i FROM 0 TO .mom$ab_service_data [svd$gk_pcno_lpc, svd$l_param] - 1 DO
;   887          0880  3     BEGIN
;   888          0881  3     !
;   889          0882  3     ! Transmit loop data and wait for completion
;   890          0883  3     !
;   891        P 0884  3     status = $QIOW (CHAN = .loop_chan,
;   892        P 0885  3                     FUNC = io$_writevblk,
;   893        P 0886  3                     IOSB = xmit_iosb,
;   894        P 0887  3                     P1   = .snddsc [1],
;   895          0888  3                     P2   = .snddsc [0]);
;   896          0889  3
;   897          0890  3     IF .status THEN status = .xmit_iosb [ios$w_status];
;   898          0891  3     !
;   899          0892  3     ! Check to see if we should log loop I/O
;   900          0893  3     !
;   901          0894  3     IF .status THEN
;   902          0895  3        mom$debug_msg (dbg$c_loopio,
;   903          0896  3                       .snddsc [1],
;   904          0897  3                       .snddsc [0],
;   905          0898  3                       $ASCID ('Test message transmitted'));
;   906          0899  3     !
;   907          0900  3     ! Map errors if there were any otherwise continue
;   908          0901  3     !
;   909          0902  3     mom_maplooperr (nma$c_sts_mld,
;   910          0903  3                     .status,
;   911          0904  3                     xmit_iosb,
;   912          0905  3                     .mom$ab_service_data [svd$gk_pcno_lpc, svd$l_param] - .i);
;   913          0906  3     !
;   914          0907  3     ! Post read to receive loop data back from mirror
;   915          0908  3     !
;   916        P 0909  3     status = $QIOW (CHAN = .loop_chan,
;   917        P 0910  3                     FUNC = io$_readvblk,
;   918        P 0911  3                     IOSB = rcv_iosb,
;   919        P 0912  3                     P1   = .rcvdsc [1],
;   920          0913  3                     P2   = .rcvdsc [0]);
;   921          0914  3
;   922          0915  3     IF .status THEN status = .rcv_iosb [ios$w_status];
;   923          0916  3
;   924          0917  3     rcv_size =.rcv_iosb [ios$w_count]; ! Set number of characters received
;   925          0918  3     !
;   926          0919  3     ! Check to see if we should log loop I/O
;   927          0920  3     !
;   928          0921  3     IF .status THEN
;   929          0922  3        mom$debug_msg (dbg$c_loopio,
;   930          0923  3                       .rcvdsc [1],
;   931          0924  3                       .rcvdsc [0],
;   932          0925  3                       $ASCID ('Test message received'));
;   933          0926  3     !
;   934          0927  3     ! Signal errors (if any).
;   935          0928  3
;   936          0929  3     mom_maplooperr (nma$c_sts_mld,
;  937          0930  3                      .status,
;  938          0931  3                      rcv_iosb,
;  939          0932  3                      .mom$ab_service_data [svd$gk_pcno_lpc, svd$l_param] - .i);
;  940          0933  3     !
;  941          0934  3     ! Check for errors on loopback.
;  942          0935  3     !
;  943          0936  3     IF NOT mom$chkbuffer (.snddsc [0],
;  944          0937  3                           .snddsc [1],
;  945          0938  3                           .rcv_size,
;  946          0939  3                           .rcvdsc [1],
;  947          0940  3                           .version) THEN
;  948          0941  4         BEGIN
;  949          0942  4         msgnotlooped = .mom$ab_service_data [svd$gk_pcno_lpc, svd$l_param] - .i; ! sET UP MESSAGES NOT LOOPE
;  950          0943  4         mom_siglooperr (nma$c_sts_blr,
;  951          0944  4                         -1,
;  952          0945  4                         UPLIT (2, msgnotlooped),
;  953          0946  4                         0,
;  954          0947  4                         msb$m_data_fld);
;  955          0948  3         END;
;  956          0949  2     END;                                 ! End of INCR loop
;  957          0950  2
;  958          0951  2 !
;  959          0952  2 ! On successful loopback, disconnect link to mirror.
;  960          0953  2 !
;  961       P  0954  2 $QIOW (CHAN = .loop_chan,
;  962          0955  2         FUNC = io$_deaccess OR io$m_synch);
;  963          0956  2 !
;  964          0957  2 ! Signal success.
;  965          0958  2 !
;  966          0959  2 mom$error (nma$c_sts_suc);
;  967          0960  2
;  968          0961  1 END;                             ! End of mom$nodetest



                                                                           .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                  0000001C  0007C P.AAL:   .LONG   28                                           ;
                                                  00000000' 00080          .ADDRESS U.3                                         ;
                                                  00000002  00084 P.AAM:   .LONG   2                                            ;
                                                  00000000' 00088          .ADDRESS BUFSIZ                                      ;
72  74  20  65  67  61  73  73  65  6D  20  74  73  65  54  0008C P.AAO:   .ASCII  \Test message transmitted\               ;
                        64  65  74  74  69  6D  73  6E  61  0009B                                                               ;
                                                  00000018  000A4 P.AAN:   .LONG   24                                           ;
                                                  00000000' 000A8          .ADDRESS P.AAO                                       ;
65  72  20  65  67  61  73  73  65  6D  20  74  73  65  54  000AC P.AAQ:   .ASCII  \Test message received\                  ;
                                    64  65  76  69  65  63  000BB                                                               ;
                                                            000C1          .BLKB   3                                            ;
                                                  00000015  000C4 P.AAP:   .LONG   21                                           ;
                                                  00000000' 000C8          .ADDRESS P.AAQ                                       ;
                                                  00000002  000CC P.AAR:   .LONG   2                                            ;
                                                  00000000' 000D0          .ADDRESS MSGNOTLOOPED                                ;

                                                                           .PSECT  $OWN$,NOEXE,2

                                                        22  001EC ; NFB
                                                                  U.3:     .BYTE   34                                           ;
```

```
                                      00  001ED          .BYTE   0                                      ;
                                      02  001EE          .BYTE   2                                      ;
                                      00  001EF          .BYTE   0                                      ;
                                02020043  001F0          .LONG   33685571                               ;
                                00000001  001F4          .LONG   1                                      ;
                                      00  001F8          .BYTE   0                                      ;
                                      00  001F9          .BYTE   0                                      ;
                                    0000  001FA          .WORD   0                                      ;
                                02010016  001FC          .LONG   33619990                               ;
                                00000000  00200          .LONG   0                                      ;
                                          00204          .BLKB   4
                                          00208 BUFSIZ:  .BLKB   4
                                          0020C MSGNOTLOOPED:
                                                         .BLKB   2

                                                U.4=                     P.AAL


                                                         .PSECT  $CODE$,NOWRT,2

                                    0FFC  00000          .ENTRY  MOM$NODETEST, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 0760
                                                                 R10,R11                                ;
               5B  00000000V   00   9E  00002          MOVAB   MOM_MAPLOOPERR, R11                      ;
               5A  00000000G   00   9E  00009          MOVAB   MOM$DEBUG_MSG, R10                       ;
               59  00000000V   00   9E  00010          MOVAB   MOM_SIGLOOPERR, R9                       ;
               58  00000000G   00   9E  00017          MOVAB   SYS$QIOW, R8                             ;
               57  00000000'   00   9E  0001E          MOVAB   NFBDSC+4, R7                             ;
               56  00000000'   00   9E  00025          MOVAB   VERSION, R6                              ;
               5E       FF70   CE   9E  0002C          MOVAB   -144(SP), SP                             ;
               6D       01D6   CF   DE  00031          MOVAL   13$, (FP)                                ; 0770
               50              67   D0  00036          MOVL    NFBDSC+4, NFB                            ; 0807
               51  00000000G   00   D0  00039          MOVL    MOM$GQ_ENTITY_BUF_DSC+4, R1              ; 0811
               01  00000000G   00   91  00040          CMPB    MOM$GB_ENTITY_CODE, #1                   ; 0808
                               11   12  00047          BNEQ    1$                                       ;
               52  00000000G   00   D0  00049          MOVL    MOM$GQ_ENTITY_BUF_DSC, ENT_LEN           ; 0810
     04        A0   02020043   8F   D0  00050          MOVL    #33685571, 4(NFB)                        ; 0812
                               0E   11  00058          BRB     2$                                       ; 0808
               52              02   CE  0005A 1$:      MNEGL   #2, ENT_LEN                              ; 0816
               51              61   D0  0005D          MOVL    (R1), ENT_ADDR                           ; 0817
     04        A0   02010012   8F   D0  00060          MOVL    #33619986, 4(NFB)                        ; 0818
               6E         68   8F   9A  00068 2$:      MOVZBL  #104, P2DSC                              ; 0820
     04        AE         08   AE   9E  0006C          MOVAB   P2NAMBUF, P2DSC+4                        ; 0821
                               5E   DD  00071          PUSHL   SP                                       ; 0822
                          04   AE   9F  00073          PUSHAB  P2DSC                                    ;
                               7E   D4  00076          CLRL    -(SP)                                    ;
               7E              01   CE  00078          MNEGL   #1, -(SP)                                ; 0824
                               51   DD  0007B          PUSHL   ENT_ADDR                                 ; 0823
                               52   DD  0007D          PUSHL   ENT_LEN                                  ; 0822
00000000G      00              06   FB  0007F          CALLS   #6, MOM$BUILD_P2                         ;
                   00000000G   00   9F  00086          PUSHAB  MOM$GQ_ACPQIO_BUF_DSC                    ; 0832
                               7E   D4  0008C          CLRL    -(SP)                                    ;
                          08   AE   9F  0008E          PUSHAB  P2DSC                                    ;
                          FC   A7   9F  00091          PUSHAB  NFBDSC                                   ;
00000000G      00              04   FB  00094          CALLS   #4, MOM$NETACP_QIO                       ;
               52              50   D0  0009B          MOVL    R0, STATUS                               ;
               50  00000000*   00   D0  0009E          MOVL    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPL*137>-; 0841
                                                               >+9>, R0                                 ;
```

```
                     14               52 E9 000A5       BLBC     STATUS, 3$                                  ; 0837
                     02  00000000G  00 D1 000A8       CMPL     MOM$AB_ACPQIO_BUFFER, #2                    ; 0838
                                      0B 12 000AF       BNEQ     3$                                          ;
                     66               02 D0 000B1       MOVL     #2, VERSION                                 ; 0840
                00E0 C6         01  A0 9E 000B4       MOVAB    1(R0), BUFSIZ                               ; 0841
                                      08 11 000BA       BRB      4$                                          ; 0837
                     66               01 D0 000BC 3$:   MOVL     #1, VERSION                                 ; 0845
                00E0 C6               50 D0 000BF       MOVL     R0, BUFSIZ                                  ; 0846
                                  78  AE 9F 000C4 4$:   PUSHAB   RCVDSC                                      ; 0851
                                  74  AE 9F 000C7       PUSHAB   SNDDSC                                      ;
                                00E0  C6 9F 000CA       PUSHAB   BUFSIZ                                      ;
           00000000V 00               03 FB 000CE       CALLS    #3, MOM$GETBUFFER                           ;
                     52               50 D0 000D5       MOVL     R0, STATUS                                  ;
                     15               52 E8 000D8       BLBS     STATUS, 5$                                  ; 0853
                                      24 DD 000DB       PUSHL    #36                                         ; 0859
                         00000000G  8F DD 000DD       PUSHL    #MOM$_ALPBFOVF                              ; 0855
                                  04  A7 9F 000E3       PUSHAB   P.AAM                                       ; 0857
                     7E         97  8F 9A 000E6       MOVZBL   #151, -(SP)                                 ; 0855
                     7E               10 CE 000EA       MNEGL    #16, -(SP)                                  ;
                     69               05 FB 000ED       CALLS    #5, MOM_SIGLOOPERR                          ;
           00000000V 00               00 FB 000F0 5$:   CALLS    #0, MOM_BLDLOOPNFB                          ; 0864
           00000000V 00               00 FB 000F7       CALLS    #0, MOM_OPENLINK                            ; 0868
                                      66 DD 000FE       PUSHL    VERSION                                     ; 0875
                         00000000*  00 DD 00100       PUSHL    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPD*137>-; 0874
                                                                 >+9>                                        ;
                                  78  AE 9F 00106       PUSHAB   SNDDSC                                      ; 0873
           00000000V 00               03 FB 00109       CALLS    #3, MOM$INITBUFFER                          ;
                     54  00000000*  00 D0 00110       MOVL     <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPC*137>-; 0879
                                                                 >+9>, R4                                    ;
                     53               01 CE 00117       MNEGL    #1, I                                       ; 0913
                                    00C3 31 0011A       BRW      10$                                         ;
                                      7E 7C 0011D 6$:   CLRQ     -(SP)                                       ; 0888
                                      7E 7C 0011F       CLRQ     -(SP)                                       ;
                                  E0  AD DD 00121       PUSHL    SNDDSC                                      ;
                                  E4  AD DD 00124       PUSHL    SNDDSC+4                                    ;
                                      7E 7C 00127       CLRQ     -(SP)                                       ;
                                  F8  AD 9F 00129       PUSHAB   XMIT_IOSB                                   ;
                                      30 DD 0012C       PUSHL    #48                                         ;
                                FED8  C6 DD 0012E       PUSHL    LOOP_CHAN                                   ;
                                      7E D4 00132       CLRL     -(SP)                                       ;
                     68               0C FB 00134       CALLS    #12, SYS$QIOW                               ;
                     52               50 D0 00137       MOVL     R0, STATUS                                  ;
                     15               52 E9 0013A       BLBC     STATUS, 7$                                  ; 0890
                     52         F8  AD 3C 0013D       MOVZWL   XMIT_IOSB, STATUS                           ;
                     0E               52 E9 00141       BLBC     STATUS, 7$                                  ; 0894
                                  24  A7 9F 00144       PUSHAB   P.AAN                                       ; 0898
                                  74  AE DD 00147       PUSHL    SNDDSC                                      ; 0897
                                  7C  AE DD 0014A       PUSHL    SNDDSC+4                                    ; 0896
                                      03 DD 0014D       PUSHL    #3                                          ; 0895
                     6A               04 FB 0014F       CALLS    #4, MOM$DEBUG_MSG                           ;
        7E 00000000* 00               53 C3 00152 7$:   SUBL3    I, <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPC*- ; 0905
                                                                 137>>+9>, -(SP)                             ;
                                  F8  AD 9F 0015A       PUSHAB   XMIT_IOSB                                   ; 0902
                                      52 DD 0015D       PUSHL    STATUS                                      ; 0903
                     7E               13 CE 0015F       MNEGL    #19, -(SP)                                  ; 0902
                     6B               04 FB 00162       CALLS    #4, MOM_MAPLOOPERR                          ;
                                      7E 7C 00165       CLRQ     -(SP)                                       ; 0913
```

```
                               7E  7C 00167         CLRQ    -(SP)                                        ;
                           E8  AD  DD 00169         PUSHL   RCVDSC                                       ;
                           EC  AD  DD 0016C         PUSHL   RCVDSC+4                                     ;
                               7E  7C 0016F         CLRQ    -(SP)                                        ;
                           F0  AD  9F 00171         PUSHAB  RCV_IOSB                                     ;
                               31  DD 00174         PUSHL   #49                                          ;
                         FED8  C6  DD 00176         PUSHL   LOOP_CHAN                                    ;
                               7E  D4 0017A         CLRL    -(SP)                                        ;
                    68         0C  FB 0017C         CALLS   #12, SYS$QIOW                                ;
                    52         50  D0 0017F         MOVL    R0, STATUS                                   ;
                    04         52  E9 00182         BLBC    STATUS, 8$                                   ; 0915
                    52     F0  AD  3C 00185         MOVZWL  RCV_IOSB, STATUS                             ;
                    55     F2  AD  3C 00189 8$:     MOVZWL  RCV_IOSB+2, RCV_SIZE                         ; 0917
                    0E         52  E9 0018D         BLBC    STATUS, 9$                                   ; 0921
                           44  A7  9F 00190         PUSHAB  P.AAP                                        ; 0925
                           7C  AE  DD 00193         PUSHL   RCVDSC                                       ; 0924
                           EC  AD  DD 00196         PUSHL   RCVDSC+4                                     ; 0923
                               03  DD 00199         PUSHL   #3                                           ; 0922
                    6A         04  FB 0019B         CALLS   #4, MOM$DEBUG_MSG                            ;
      7E 00000000*  00         53  C3 0019E 9$:     SUBL3   I, <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPC*-  ; 0932
                                                            137>>+9>, -(SP)                              ;
                           F0  AD  9F 001A6         PUSHAB  RCV_IOSB                                     ; 0929
                               52  DD 001A9         PUSHL   STATUS                                       ; 0930
                    7E         13  CE 001AB         MNEGL   #19, -(SP)                                   ; 0929
                    6B         04  FB 001AE         CALLS   #4, MOM_MAPLOOPERR                           ;
                               66  DD 001B1         PUSHL   VERSION                                      ; 0940
                           EC  AD  DD 001B3         PUSHL   RCVDSC+4                                     ; 0939
                               55  DD 001B6         PUSHL   RCV_SIZE                                     ; 0938
                    7E     E0  AD  7D 001B8         MOVQ    SNDDSC, -(SP)                                ; 0936
         00000000V  00         05  FB 001BC         CALLS   #5, MOM$CHKBUFFER                            ;
                    1A         50  E8 001C3         BLBS    R0, 10$                                      ;
00E4  C6 00000000*  00         53  A3 001C6         SUBW3   I, <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPC*-  ; 0942
                                                            137>>+9>, MSGNOTLOOPED                       ;
                               20  DD 001D0         PUSHL   #32                                          ; 0943
                               7E  D4 001D2         CLRL    -(SP)                                        ;
                           4C  A7  9F 001D4         PUSHAB  P.AAR                                        ; 0945
                    7E         01  CE 001D7         MNEGL   #1, -(SP)                                    ; 0944
                    7E         1C  CE 001DA         MNEGL   #28, -(SP)                                   ; 0943
                    69         05  FB 001DD         CALLS   #5, MOM_SIGLOOPERR                           ;
         02         53         54  F2 001E0 10$:    AOBLSS  R4, I, 11$                                   ; 0879
                               03  11 001E4         BRB     12$                                          ;
                             FF34  31 001E6 11$:    BRW     6$                                           ;
                               7E  7C 001E9 12$:    CLRQ    -(SP)                                        ; 0955
                               7E  7C 001EB         CLRQ    -(SP)                                        ;
                               7E  7C 001ED         CLRQ    -(SP)                                        ;
                               7E  7C 001EF         CLRQ    -(SP)                                        ;
                               7E  D4 001F1         CLRL    -(SP)                                        ;
                    7E   0234  8F  3C 001F3         MOVZWL  #564, -(SP)                                  ;
                         FED8  C6  DD 001F8         PUSHL   LOOP_CHAN                                    ;
                               7E  D4 001FC         CLRL    -(SP)                                        ;
                    68         0C  FB 001FE         CALLS   #12, SYS$QIOW                                ;
                               01  DD 00201         PUSHL   #1                                           ; 0959
         00000000G  00         01  FB 00203         CALLS   #1, MOM$ERROR                                ;
                                   04 0020A         RET                                                  ; 0961
                                 0000 0020B 13$:    .WORD   Save nothing                                 ; 0770
                               7E  D4 0020D         CLRL    -(SP)                                        ;
                               5E  DD 0020F         PUSHL   SP                                           ;
```

```
                              7E         04    AC  7D 00211           MOVQ      4(AP), -(SP)                                     ;
                       00000000V  00           03  FB 00215           CALLS     #3, MOM$TESTHANDLER                              ;
                                                   04 0021C           RET                                                        ;

; Routine Size:  541 bytes,     Routine Base:  $CODE$ + 04A6
;  970         0962  1 %SBTTL 'mom$initbuffer  Initialize loopback test buffer'
;  971         0963  1 GLOBAL ROUTINE mom$initbuffer (bufdsc, data, type) : NOVALUE =
;  972         0964  1
;  973         0965  1 !++
;  974         0966  1 ! FUNCTIONAL DESCRIPTION:
;  975         0967  1 !
;  976         0968  1 !       This routine initializes a transmit buffer for active loop
;  977         0969  1 !       functions based on the type of loopback (phase 2, phase 3, MOP)
;  978         0970  1 !       and based on the type of data (ones, zeros, mixed).
;  979         0971  1 !
;  980         0972  1 ! FORMAL PARAMETERS:
;  981         0973  1 !
;  982         0974  1 !       BUFDSC          Address of descriptor of buffer to be initialized.
;  983         0975  1 !       DATA            Type of data to use.
;  984         0976  1 !       TYPE            Loopback function type code.
;  985         0977  1 !
;  986         0978  1 !--
;  987         0979  1
;  988         0980  2 BEGIN
;  989         0981  2
;  990         0982  2 MAP
;  991         0983  2     bufdsc : REF VECTOR;
;  992         0984  2
;  993         0985  2 !
;  994         0986  2 ! Macro to add a "forward" header to an NI loop message.
;  995         0987  2 !
;  996         0988  2 MACRO
;  997         0989  2     $mom_forward_msg (msg_ptr,
;  998         0990  2                         ni_addr_svd,
;  999         0991  2                         node_addr_svd,
; 1000         0992  2                         node_name_svd,
; 1001       M 0993  2                         ni_hardw_svd) =
; 1002       M 0994  2         BEGIN
; 1003       M 0995  2         CH$WCHAR_A (mop$c_niloop_forward, msg_ptr);
; 1004       M 0996  2         CH$WCHAR_A (0, msg_ptr);
; 1005       M 0997  2
; 1006       M 0998  2     %IF NOT %NULL (ni_addr_svd)
; 1007       M 0999  2     %THEN
; 1008       M 1000  2         IF .mom$ab_service_data [ni_addr_svd, svd$v_msg_param] THEN
; 1009       M 1001  2             msg_ptr = CH$MOVE (6,
; 1010       M 1002  2                             mom$ab_service_data [ni_addr_svd, svd$t_string],
; 1011       M 1003  2                             .msg_ptr)
; 1012       M 1004  2         ELSE
; 1013       M 1005  2     %FI
; 1014       M 1006  2             BEGIN
; 1015       M 1007  2             mom$get_node_id (node_addr_svd,
; 1016       M 1008  2                                node_name_svd,
; 1017       M 1009  2                                ni_hardw_svd);
; 1018       M 1010  2             .msg_ptr = mom$k_ni_prefix;
; 1019       M 1011  2             msg_ptr = .msg_ptr + 4;
; 1020       M 1012  2             msg_ptr = CH$MOVE (2,
; 1021       M 1013  2                                 mom$ab_service_data [node_addr_svd,
; 1022       M 1014  2                                                           svd$l_param],
; 1023       M 1015  2                                 .msg_ptr);
; 1024       M 1016  2             END;
; 1025       M 1017  2         END
; 1026         1018  2         %;
; 1027          1019  2
; 1028          1020  2  LOCAL
; 1029          1021  2      loopdata : BYTE,
; 1030          1022  2      ptr;
; 1031          1023  2
; 1032          1024  2  ptr = .bufdsc [1];
; 1033          1025  2  !
; 1034          1026  2  ! Fill in the function code according to the type of loop operation.
; 1035          1027  2  !
; 1036          1028  2  SELECTONEU .type OF
; 1037          1029  2      SET
; 1038          1030  2      [mom$c_loop_phase2]:
; 1039          1031  3          BEGIN
; 1040          1032  3          CH$WCHAR_A (5, ptr);
; 1041          1033  3          CH$WCHAR_A (0, ptr);
; 1042          1034  2          END;
; 1043          1035  2
; 1044          1036  2      [mom$c_loop_phase3]:
; 1045          1037  2          CH$WCHAR_A (0, ptr);
; 1046          1038  2
; 1047          1039  2      [mom$c_loop_mop]:
; 1048          1040  3          BEGIN
; 1049          1041  3          IF NOT .mom$gl_service_flags [mom$v_ni_circ] THEN
; 1050          1042  3              CH$WCHAR_A (mop$_fct_ald, ptr)
; 1051          1043  3          ELSE
; 1052          1044  3              !
; 1053          1045  3              ! The circuit being looped is an NI.  The format if these loop
; 1054          1046  3              ! messages is completely different because the looping is done
; 1055          1047  3              ! by the NIs UNA device.
; 1056          1048  3              !
; 1057          1049  4              BEGIN
; 1058          1050  4              ptr = CH$FILL (0, 2, .ptr);          ! Skip count
; 1059          1051  4              SELECT .mom$ab_service_data [svd$gk_pcno_lph, svd$l_param] OF
; 1060          1052  4                  SET
; 1061          1053  4                  [nma$c_loop_full,nma$c_loop_xmit]:
; 1062          1054  4                      !
; 1063          1055  4                      ! The loop message will be sent to the assist NI
; 1064          1056  4                      ! address first.  From there, it must be forwarded
; 1065          1057  4                      ! to the target.  Add "forward" message header to
; 1066          1058  4                      ! loop message so the assist node will send the
; 1067          1059  4                      ! message on to the target node.
; 1068          1060  4                      !
; 1069        P 1061  4                      $mom_forward_msg (ptr, svd$gk_pcno_pha,
; 1070        P 1062  4                                          svd$gk_pcno_lpn,
; 1071        P 1063  4                                          svd$gk_pcno_$lna,
; 1072          1064  4                                          svd$gk_pcno_$lnh);
; 1073          1065  4                  [nma$c_loop_full,nma$c_loop_recv]:
; 1074          1066  4                      !
; 1075          1067  4                      ! From the target node, the message must be forwarded to
; 1076          1068  4                      ! the receive assist node.  Add the "forward" header to
; 1077          1069  4                      ! accomplish this.
; 1078          1070  4                      !
; 1079        P 1071  4                      $mom_forward_msg (ptr, svd$gk_pcno_lpa,
; 1080        P 1072  4                                          svd$gk_pcno_lan,
; 1081        P 1073  4                                          svd$gk_pcno_$lnn,
; 1082          1074  4                                          svd$gk_pcno_$lah);
; 1083          1075  4                  [ALWAYS]:
```

```
; 1084          1076  4                                 !
; 1085          1077  4                                 ! The message must be returned to this node.  Add a
; 1086          1078  4                                 ! "forward" header to the loop message which contains my
; 1087          1079  4                                 ! own NI address.
; 1088          1080  4                                 !
; 1089        P 1081  4                                 $mom_forward_msg (ptr,,
; 1090        P 1082  4                                                         svd$gk_pcno_iho,
; 1091        P 1083  4                                                         svd$gk_pcno_$hna,
; 1092          1084  4                                                         svd$gk_pcno_$hhw);
; 1093          1085  4                             TES;
; 1094          1086  4                         !
; 1095          1087  4                         ! Now add the "reply" message function and receipt number.
; 1096          1088  4                         !
; 1097          1089  4                         CH$WCHAR_A (mop$c_niloop_reply, ptr);
; 1098          1090  4                         CH$WCHAR_A (0, ptr);
; 1099          1091  4                         ptr = CH$FILL (0, 2, .ptr),             ! Receipt number
; 1100          1092  3                         END;
; 1101          1093  2                     END;
; 1102          1094  2                 TES;
; 1103          1095  2             !
; 1104          1096  2             ! Select type of data to put in buffer.
; 1105          1097  2             !
; 1106          1098  2             SELECTONEU .data OF
; 1107          1099  2                 SET
; 1108          1100  2                 [nma$c_loop_mix]: loopdata = %B'01010101';
; 1109          1101  2                 [nma$c_loop_one]: loopdata = %X'FF';
; 1110          1102  2                 [nma$c_loop_zer]: loopdata = 0;
; 1111          1103  2                 TES;
; 1112          1104  2             !
; 1113          1105  2             ! Initialize the buffer.
; 1114          1106  2             !
; 1115          1107  2             CH$FILL (.loopdata,
; 1116          1108  2                      .bufdsc [0] - (.ptr - .bufdsc [1]),
; 1117          1109  2                      .ptr);
; 1118          1110  2
; 1119          1111  1             END;                                    ! End of mom$initbuffer




                                         01FC 00000             .ENTRY   MOM$INITBUFFER, Save R2,R3,R4,R5,R6,R7,R8     ; 0963
                        58 0000000G   00   9E 00002             MOVAB    MOM$GET_NODE_ID, R8                           ;
                        57        04  AC   D0 00009             MOVL     BUFDSC, R7                                    ; 1024
                        53        04  A7   D0 0000D             MOVL     4(R7), PTR                                    ;
                        50        0C  AC   D0 00011             MOVL     TYPE, R0                                      ; 1028
                        02            50   D1 00015             CMPL     R0, #2                                        ; 1030
                                      05   12 00018             BNEQ     1$                                            ;
                        83            05   90 0001A             MOVB     #5, (PTR)+                                    ; 1032
                                      05   11 0001D             BRB      2$                                            ; 1033
                        01            50   D1 0001F 1$:         CMPL     R0, #1                                        ; 1036
                                      04   12 00022             BNEQ     3$                                            ;
                                      63   94 00024 2$:         CLRB     (PTR)                                         ; 1037
                                      0F   11 00026             BRB      4$                                            ;
                                      50   D5 00028 3$:         TSTL     R0                                            ; 1039
                                      0D   12 0002A             BNEQ     5$                                            ;
                08 0000000G   00      01   E0 0002C             BBS      #1, MOM$GL_SERVICE_FLAGS, 6$                  ; 1041
```

```
                   63                18  90 00034          MOVB    #24, (PTR)                                          ; 1042
                                     53  D6 00037 4$:      INCL    PTR                                                 ;
                                   00B0  31 00039 5$:      BRW     12$                                                 ;
                                     83  B4 0003C 6$:      CLRW    (PTR)+                                              ; 1050
                   56 00000000*      00  D0 0003E          MOVL    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPH*137>-;       1051
                                                                   >+9>, R6                                            ;
                                     05  13 00045          BEQL    7$                                                  ; 1053
                   02                56  D1 00047          CMPL    R6, #2                                              ;
                                     37  12 0004A          BNEQ    9$                                                  ;
                   83                02  B0 0004C 7$:      MOVW    #2, (PTR)+                                          ; 1064
                   0A 00000000*      00  E9 0004F          BLBC    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_PHA*137>-;
                                                                   >+7>, 8$                                            ;
        63 00000000*  00             06  28 00056          MOVC3   #6, <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_PHA*-;
                                                                   137>>+9>, (PTR)                                     ;
                                     23  11 0005E          BRB     9$                                                  ;
                      00000000G      8F  DD 00060 8$:      PUSHL   #SVD$GK_PCNO_$LNH                                   ;
                      00000000G      8F  DD 00066          PUSHL   #SVD$GK_PCNO_$LNA                                   ;
                      00000000G      8F  DD 0006C          PUSHL   #SVD$GK_PCNO_LPN                                    ;
                   68                03  FB 00072          CALLS   #3, MOM$GET_NODE_ID                                 ;
                   83 000400AA       8F  D0 00075          MOVL    #262314, (PTR)+                                     ;
                   83 00000000*      00  B0 0007C          MOVW    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPN*137>-;
                                                                   >+9>, (PTR)+                                        ;
                                     56  D5 00083 9$:      TSTL    R6                                                  ; 1065
                                     3C  15 00085          BLEQ    11$                                                 ;
                   02                56  D1 00087          CMPL    R6, #2                                              ;
                                     37  14 0008A          BGTR    11$                                                 ;
                   83                02  B0 0008C          MOVW    #2, (PTR)+                                          ; 1074
                   0A 00000000*      00  E9 0008F          BLBC    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPA*137>-;
                                                                   >+7>, 10$                                           ;
        63 00000000*  00             06  28 00096          MOVC3   #6, <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPA*-;
                                                                   137>>+9>, (PTR)                                     ;
                                     23  11 0009E          BRB     11$                                                 ;
                      00000000G      8F  DD 000A0 10$:     PUSHL   #SVD$GK_PCNO_$LAH                                   ;
                      00000000G      8F  DD 000A6          PUSHL   #SVD$GK_PCNO_$LNN                                   ;
                      00000000G      8F  DD 000AC          PUSHL   #SVD$GK_PCNO_LAN                                    ;
                   68                03  FB 000B2          CALLS   #3, MOM$GET_NODE_ID                                 ;
                   83 000400AA       8F  D0 000B5          MOVL    #262314, (PTR)+                                     ;
                   83 00000000*      00  B0 000BC          MOVW    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LAN*137>-;
                                                                   >+9>, (PTR)+                                        ;
                   83                02  B0 000C3 11$:     MOVW    #2, (PTR)+                                          ; 1084
                      00000000G      8F  DD 000C6          PUSHL   #SVD$GK_PCNO_$HHW                                   ;
                      00000000G      8F  DD 000CC          PUSHL   #SVD$GK_PCNO_$HNA                                   ;
                      00000000G      8F  DD 000D2          PUSHL   #SVD$GK_PCNO_IHO                                    ;
                   68                03  FB 000D8          CALLS   #3, MOM$GET_NODE_ID                                 ;
                   83 000400AA       8F  D0 000DB          MOVL    #262314, (PTR)+                                     ;
                   83 00000000*      00  B0 000E2          MOVW    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_IHO*137>-;
                                                                   >+9>, (PTR)+                                        ;
                   83                01  D0 000E9          MOVL    #1, (PTR)+                                          ; 1089
                   50          08    AC  D0 000EC 12$:     MOVL    DATA, R0                                            ; 1098
                   02                50  D1 000F0          CMPL    R0, #2                                              ; 1100
                                     06  12 000F3          BNEQ    13$                                                 ;
                   51          55    8F  90 000F5          MOVB    #85, LOOPDATA                                       ;
                                     10  11 000F9          BRB     15$                                                 ;
                   01                50  D1 000FB 13$:     CMPL    R0, #1                                              ; 1101
                                     05  12 000FE          BNEQ    14$                                                 ;
                   51                01  8E 00100          MNEGB   #1, LOOPDATA                                        ;
                                     06  11 00103          BRB     15$                                                 ;
```

```
                                             50  D5 00105 14$:    TSTL    R0                                              ; 1102
                                             02  12 00107         BNEQ    15$                                             :
                                             51  94 00109         CLRB    LOOPDATA                                        :
                  50        04      A7       53  C3 0010B 15$:    SUBL3   PTR, 4(R7), R0                                  ; 1108
                                    50       67  C0 00110         ADDL2   (R7), R0                                        :
         50       51                6E       00  2C 00113         MOVC5   #0, (SP), LOOPDATA, R0, (PTR)                   ; 1109
                                             63     00118                                                                 :
                                                 04 00119         RET                                                     ; 1111

; Routine Size:  282 bytes,    Routine Base:  $CODE$ + 06C3
```

```
1 %SBTTL 'mom$chkbuffer  Check loopback test buffer'
1 GLOBAL ROUTINE mom$chkbuffer (sndlen, sndadr, rcvlen, rcvadr, type) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       SNDLEN          Length of transmitted data.
1 !       SNDADR          Address of transmitted data.
1 !       RCVLEN          Length of received data.
1 !       RCVADR          Address of received data.
1 !       TYPE            Loopback function type code.
1 !
1 !--
1
2 BEGIN
2
2 LOCAL
2     rcvptr,
2     sndptr,
2     skip_count,
2     status;
2
2 rcvptr = .rcvadr;
2 sndptr = .sndadr + 1;
2
2 SELECTU .type OF
2     SET
2     [mom$c_loop_phase2
2     ,mom$c_loop_phase3]:
2         IF CH$RCHAR_A (rcvptr) NEQU 1 THEN
2             RETURN false;
2
2     [mom$c_loop_phase2]:
3         BEGIN
3         CH$RCHAR_A (sndptr);
3         CH$RCHAR_A (rcvptr);
2         END;
2
2     [mom$c_loop_mop]:
3         BEGIN
3         IF NOT .mom$gl_service_flags [mom$v_ni_circ] THEN
4             BEGIN
4             IF CH$RCHAR (.rcvptr) NEQU mop$_fct_ald
4                     AND
4                CH$RCHAR (.rcvptr) NEQU mop$_fct_pld THEN
4                 RETURN false;
4             CH$RCHAR_A (rcvptr);
4             END
3         ELSE
3             !
3             ! The message was looped on the NI, so the format is different.
3             !
4             BEGIN
4             skip_count = .(.rcvptr)<0,16> + 2;
```

```
; 1178       1169 4                rcvptr = .rcvptr + .skip_count;
; 1179       1170 4                IF CH$RCHAR (.rcvptr) NEQU mop$c_niloop_reply THEN
; 1180       1171 4                    RETURN false;
; 1181       1172 4                !
; 1182       1173 4                ! For now, ignore the receipt number.  So, skip the "reply"
; 1183       1174 4                ! function code and the receipt number;
; 1184       1175 4                !
; 1185       1176 4                rcvptr = .rcvptr + 4;
; 1186       1177 4                sndptr = .sndadr + .skip_count + 4;
; 1187       1178 3                END;
; 1188       1179 2            END;
; 1189       1180 2        TES;
; 1190       1181 2
; 1191       1182 2    IF CH$EQL ( .rcvlen - (.rcvptr - .rcvadr),
; 1192       1183 2                .rcvptr,
; 1193       1184 2                .sndlen - (.sndptr - .sndadr),
; 1194       1185 2                .sndptr) THEN
; 1195       1186 2        RETURN true
; 1196       1187 2    ELSE
; 1197       1188 2        RETURN false;
; 1198       1189 2
; 1199       1190 1    END;                                    ! End of mom$chkbuffer




                                      000C 00000           .ENTRY  MOM$CHKBUFFER, Save R2,R3                        ; 1113
                   53        10    AC   D0 00002           MOVL    RCVADR, RCVPTR                                   ; 1137
    50        08   AC              01   C1 00006           ADDL3   #1, SNDADR, SNDPTR                               ; 1138
                   51        14    AC   D0 0000B           MOVL    TYPE, R1                                         ; 1140
                                   0D   13 0000F           BEQL    1$                                               ; 1142
                   02              51   D1 00011           CMPL    R1, #2                                           ;
                                   08   1A 00014           BGTRU   1$                                               ;
                   52              83   9A 00016           MOVZBL  (RCVPTR)+, R2                                    ; 1144
                   01              52   91 00019           CMPB    R2, #1                                           ;
                                   5A   12 0001C           BNEQ    6$                                               ;
                   02              51   D1 0001E 1$:       CMPL    R1, #2                                           ; 1147
                                   04   12 00021           BNEQ    2$                                               ;
                                   50   D6 00023           INCL    SNDPTR                                           ; 1149
                                   53   D6 00025           INCL    RCVPTR                                           ; 1150
                                   51   D5 00027 2$:       TSTL    R1                                               ; 1153
                                   2F   12 00029           BNEQ    5$                                               ;
    0E 00000000G   00              01   E0 0002B           BBS     #1, MOM$GL_SERVICE_FLAGS, 4$                     ; 1155
                   18              63   91 00033           CMPB    (RCVPTR), #24                                    ; 1157
                                   05   13 00036           BEQL    3$                                               ;
                   19              63   91 00038           CMPB    (RCVPTR), #25                                    ; 1159
                                   3B   12 0003B           BNEQ    6$                                               ;
                                   53   D6 0003D 3$:       INCL    RCVPTR                                           ; 1161
                                   19   11 0003F           BRB     5$                                               ; 1155
                   51              63   3C 00041 4$:       MOVZWL  (RCVPTR), SKIP_COUNT                             ; 1168
                   51              02   C0 00044           ADDL2   #2, SKIP_COUNT                                   ;
                   53              51   C0 00047           ADDL2   SKIP_COUNT, RCVPTR                               ; 1169
                   01              63   91 0004A           CMPB    (RCVPTR), #1                                     ; 1170
                                   29   12 0004D           BNEQ    6$                                               ;
                   53              04   C0 0004F           ADDL2   #4, RCVPTR                                       ; 1176
                   51        08    AC   C0 00052           ADDL2   SNDADR, R1                                       ; 1177
```

```
                                  50        04    A1   9E 00056           MOVAB    4(R1), SNDPTR                                 ;
                   52        10   AC              53   C3 0005A 5$:       SUBL3    RCVPTR, RCVADR, R2                            ; 1182
                                  52        0C    AC   C0 0005F           ADDL2    RCVLEN, R2                                    ;
                   51        08   AC              50   C3 00063           SUBL3    SNDPTR, SNDADR, R1                            ; 1184
                                  51        04    AC   C0 00068           ADDL2    SNDLEN, R1                                    ;
        51         00             63              52   2D 0006C           CMPC5    R2, (RCVPTR), #0, R1, (SNDPTR)                ; 1182
                                                  60      00071                                                                  ;
                                                  04   12 00072           BNEQ     6$                                            ;
                         50                       01   D0 00074           MOVL     #1, R0                                        ; 1188
                                                       04 00077           RET                                                    ;
                                            50         D4 00078 6$:       CLRL     R0                                            ; 1190
                                                       04 0007A           RET                                                    ;

; Routine Size:  123 bytes,    Routine Base:  $CODE$ + 07DD
```

```
1 %SBTTL 'mom$getbuffer  Allocate send/receive buffers for loopback'
1 GLOBAL ROUTINE mom$getbuffer (reqsiz, rbfdsc, sbfdsc) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine will dynamically allocate buffers for loopback.
1 !
1 !       If there is not enough virtual memory then this routine will
1 !       return an error and set the maximum size into the descriptors.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       REQSIZ          Address of longword size of buffer to allocate.
1 !       RBFDSC          Address of receive buffer descriptor.
1 !       SBFDSC          Address of send buffer descriptor.
1 !
1 !--
2 BEGIN
2
2 MAP
2     rbfdsc : REF VECTOR,
2     sbfdsc : REF VECTOR;
2
2 LOCAL
2     status;
2
2 mom$l_vmbufsize = 0;                ! No buffer allocated
2 rbfdsc [1] = mom$t_testrcvbuf;
2 sbfdsc [1] = mom$ab_nice_xmit_buf;
2 !
2 ! Determine if we need a buffer bigger then the default buffer.
2 !
2 IF ..reqsiz LEQU mom$k_maxmsgsize THEN
3     BEGIN
3     rbfdsc [0]  = ..reqsiz;
3     sbfdsc [0]  = ..reqsiz;
3     RETURN true
3     END;
2
2 mom$l_vmbufsize = ..reqsiz * 2; ! Set up allocation size
2 !
2 ! Attempt to get the necessary size buffer.
2 !
2 status = LIB$GET_VM (mom$l_vmbufsize, mom$l_vm_buf_adr);
2
2 IF .status THEN
3     BEGIN
3     rbfdsc [1] = .mom$l_vm_buf_adr;
3     sbfdsc [1] = .mom$l_vm_buf_adr + (.mom$l_vmbufsize / 2);
3     END
2 ELSE
2     .reqsiz = mom$k_defbufsize;
2
2 rbfdsc [0]  = ..reqsiz;
2 sbfdsc [0]  = ..reqsiz;
2
```

```
; 1258          1248   2 RETURN .status
; 1259          1249   2
; 1260          1250   1 END;                                   ! End of mom$getbuffer



                                                003C 00000             .ENTRY   MOM$GETBUFFER, Save R2,R3,R4,R5                           ; 1192
                        55 00000000'      00     9E 00002             MOVAB    MOM$L_VMBUFSIZE, R5                                       ;
                                          65     D4 00009             CLRL     MOM$L_VMBUFSIZE                                           ; 1218
                        53        08      AC     D0 0000B             MOVL     RBFDSC, R3                                                ; 1219
                  04    A3        08      A5     9E 0000F             MOVAB    MOM$T_TESTRCVBUF, 4(R3)                                   ;
                        52        0C      AC     D0 00014             MOVL     SBFDSC, R2                                                ; 1220
                  04    A2 00000000G      00     9E 00018             MOVAB    MOM$AB_NICE_XMIT_BUF, 4(R2)                               ;
            00000080    8F        04      BC     D1 00020             CMPL     @REQSIZ, #128                                             ; 1224
                                          0C     1A 00028             BGTRU    1$                                                        ;
                        63        04      BC     D0 0002A             MOVL     @REQSIZ, (R3)                                             ; 1226
                        62        04      BC     D0 0002E             MOVL     @REQSIZ, (R2)                                             ; 1227
                        50                01     D0 00032             MOVL     #1, R0                                                    ; 1228
                                                 04 00035             RET                                                                ;
         65       04    BC                01     78 00036 1$:         ASHL     #1, @REQSIZ, MOM$L_VMBUFSIZE                              ; 1231
                                  04      A5     9F 0003B             PUSHAB   MOM$L_VM_BUF_ADR                                          ; 1235
                                          55     DD 0003E             PUSHL    R5                                                        ;
            00000000G   00                02     FB 00040             CALLS    #2, LIB$GET_VM                                            ;
                        13                50     E9 00047             BLBC     STATUS, 2$                                                ; 1237
                        54        04      A5     D0 0004A             MOVL     MOM$L_VM_BUF_ADR, R4                                      ; 1239
                  04    A3                54     D0 0004E             MOVL     R4, 4(R3)                                                 ;
         51             65                02     C7 00052             DIVL3    #2, MOM$L_VMBUFSIZE, R1                                   ; 1240
   04    A2             51                54     C1 00056             ADDL3    R4, R1, 4(R2)                                             ;
                                          05     11 0005B             BRB      3$                                                        ; 1237
                  04    BC        82      8F     9A 0005D 2$:         MOVZBL   #130, @REQSIZ                                             ; 1243
                        63        04      BC     D0 00062 3$:         MOVL     @REQSIZ, (R3)                                             ; 1245
                        62        04      BC     D0 00066             MOVL     @REQSIZ, (R2)                                             ; 1246
                                                 04 0006A             RET                                                                ; 1250

; Routine Size:  107 bytes,    Routine Base:  $CODE$ + 0858
; 1262         1251  1 %SBTTL 'mom$freebuffer  Deallocate buffers'
; 1263         1252  1 GLOBAL ROUTINE mom$freebuffer =
; 1264         1253  1
; 1265         1254  1 !++
; 1266         1255  1 ! FUNCTIONAL DESCRIPTION:
; 1267         1256  1 !
; 1268         1257  1 !       This routine deallocates buffers if any were allocated by the
; 1269         1258  1 !       MOM$GETBUFFER routine.
; 1270         1259  1 !
; 1271         1260  1 ! IMPLICIT INPUTS:
; 1272         1261  1 !
; 1273         1262  1 !       MOM$L_VMBUFSIZE contains the size of the allocated buffers.
; 1274         1263  1 !           If none were allocated then the value will be zero.
; 1275         1264  1 !
; 1276         1265  1 !--
; 1277         1266  2 BEGIN
; 1278         1267  2
; 1279         1268  2 LOCAL
; 1280         1269  2     status;
; 1281         1270  2
; 1282         1271  2 IF .mom$l_vmbufsize NEQ 0 THEN
; 1283         1272  2     status = LIB$FREE_VM (mom$l_vmbufsize, mom$l_vm_buf_adr);
; 1284         1273  2
; 1285         1274  2 mom$l_vmbufsize = 0;                ! No buffer allocated
; 1286         1275  2
; 1287         1276  2 RETURN .status
; 1288         1277  2
; 1289         1278  1 END;                                ! End of mom$freebuffer



                                   0004 00000         .ENTRY  MOM$FREEBUFFER, Save R2                              ; 1252
              52 00000000'   00      9E 00002         MOVAB   MOM$L_VMBUFSIZE, R2                                  ;
                             62      D5 00009         TSTL    MOM$L_VMBUFSIZE                                      ; 1271
                             0C      13 0000B         BEQL    1$                                                   ;
                      04     A2      9F 0000D         PUSHAB  MOM$L_VM_BUF_ADR                                     ; 1272
                             52      DD 00010         PUSHL   R2                                                   ;
         00000000G  00       02      FB 00012         CALLS   #2, LIB$FREE_VM                                      ;
                             62      D4 00019 1$:     CLRL    MOM$L_VMBUFSIZE                                      ; 1274
                                     04 0001B         RET                                                          ; 1278

; Routine Size:  28 bytes,    Routine Base:  $CODE$ + 08C3
```

```
    1 %SBTTL 'mom_openlink  Open a link to Mirror'
    1 ROUTINE mom_openlink : NOVALUE =
    1
    1 !++
    1 ! FUNCTIONAL DESCRIPTION:
    1 !
    1 !       This routine opens a link to the loopback mirror and
    1 !       verifies the connect data to determine the maximum test message size.
    1 !
    1 !--
    1
    2 BEGIN
    2
    2 OWN
    2     mbx_maxmsg : INITIAL (64),              ! Maximum mailbox message size
    2     mbx_bufquo : INITIAL (256);             ! Maximum pool for mailbox messages
    2
    2 LOCAL
    2     mbxname      : VECTOR [10, BYTE],       ! Buffer to build mailbox name
    2     mbxlst       : VECTOR [2],              ! FAO list for mailbox name
    2     mbxdsc       : VECTOR [2],              ! Descriptor of mailbox name buffer
    2     iosb         : $iosb,                   ! IO status block
    2     status,                                 ! Return status
    2     ptr,                                    ! General pointer
    2     ctr,                                    ! General counter
    2     detail       : WORD,                    ! Error detail word
    2     flag,
    2     text,
    2     code         : BYTE,                    ! Error status code
    2     loop_descr : VECTOR [2],                ! Loop message length descriptor
    2     mirror_descr : VECTOR [2];              ! Mirror size descriptor
    2
    2 OWN
    2     chnchar : BBLOCK [dib$k_length]; ! Channel characteristics
    2
    2 !
    2 ! Establish a channel to looper object
    2 !
    2 status = LIB$ASN_WTH_MBX
    2                   (
    2                   mom$gq_netnamdsc, ! Device name
    2                   mbx_maxmsg,       ! Maximum mailbox message size
    2                   mbx_bufquo,       ! Maximum buffer quota for mailbox I/O
    2                   loop_chan,        ! I/O channel for mirror
    2                   loop_mbxchan      ! Mailbox channel for mirror
    2                   );
    2 !
    2 ! Map and signal any errors
    2 !
    2 mom_maplooperr (nma$c_sts_mcf, .status, 0, .mom$ab_service_data [svd$gk_pcno_lpc, svd$l_param]);
    2
P   2 status = $QIOW                        ! Create a logical link to mirror
P   2             (
P   2             CHAN = .loop_chan,    ! Use network channel
P   2             FUNC = io$_access,    ! ACP function
P   2             IOSB = iosb,          ! Status here
P   2             P2   = nfbdsc         ! This is the NFB descriptor
; 1348            1336  2                );
; 1349            1337  2     !
; 1350            1338  2     ! Map and signal any errors
; 1351            1339  2     !
; 1352            1340  2     mom_maplooperr (nma$c_sts_mcf, .status, iosb,
; 1353            1341  2                     .mom$ab_service_data [svd$gk_pcno_lpc, svd$l_param]);
; 1354            1342  2
; 1355          P 1343  2     status = $QIOW                  ! Read the connect data
; 1356          P 1344  2             (
; 1357          P 1345  2             CHAN = .loop_mbxchan,   ! Channel for mailbox
; 1358          P 1346  2             FUNC = io$_readvblk,
; 1359          P 1347  2             IOSB = iosb,
; 1360          P 1348  2             P1   = mbx_buffer,      ! Read data into mailbox buffer
; 1361          P 1349  2             P2   = mbx_size
; 1362            1350  2             );
; 1363            1351  2     !
; 1364            1352  2     ! Map and signal any errors
; 1365            1353  2     !
; 1366            1354  2     mom_maplooperr (nma$c_sts_mcf, .status, iosb,
; 1367            1355  2                     .mom$ab_service_data [svd$gk_pcno_lpc, svd$l_param]);
; 1368            1356  2     !
; 1369            1357  2     ! Validate the mailbox message and its returned optional data.
; 1370            1358  2     !
; 1371            1359  2     status = .(mbx_buffer)<0,16,0>;
; 1372            1360  2     ptr = mbx_buffer + 4;
; 1373            1361  2
; 1374            1362  2     IF .status NEQ msg$_confirm THEN           ! It must be a connect confirm
; 1375            1363  3         BEGIN
; 1376            1364  3         flag = 0;
; 1377            1365  3         code = nma$c_sts_mcf;                  ! Assume connect failure code
; 1378            1366  3         text = 0;
; 1379            1367  3
; 1380            1368  3         SELECTONE  .status OF
; 1381            1369  3             SET
; 1382            1370  3             [msg$_abort
; 1383            1371  3             ,msg$_exit]:
; 1384            1372  3                 detail = nma$c_ncedtl_abo; ! Abort by object
; 1385            1373  3
; 1386            1374  3             [msg$_discon]:
; 1387            1375  3                 detail = nma$c_ncedtl_dis; ! Disconnect by object
; 1388            1376  3
; 1389            1377  3             [msg$_netshut]:
; 1390            1378  3                 detail = nma$c_ncedtl_nsd; ! Node shut down
; 1391            1379  3
; 1392            1380  3             [msg$_nodeinacc
; 1393            1381  3             ,msg$_pathlost]:
; 1394            1382  3                 detail = nma$c_ncedtl_die; ! Node or object failed
; 1395            1383  3
; 1396            1384  3             [msg$_reject]:
; 1397            1385  3                 detail = nma$c_ncedtl_rjc; ! Reject by object
; 1398            1386  3
; 1399            1387  3             [msg$_thirdparty]:
; 1400            1388  3                 detail = nma$c_ncedtl_abm; ! Abort by management
; 1401            1389  3
; 1402            1390  3             [msg$_timeout]:
; 1403            1391  3                 detail  = nma$c_ncedtl_nrs; ! No response from object
; 1404            1392  3
```

```
: 1405          1393  3             [OTHERWISE]:
: 1406          1394  4                 BEGIN
: 1407          1395  4                 detail = -1;
: 1408          1396  4                 flag = .flag OR msb$m_msg_fld;
: 1409          1397  4                 text = .status;
: 1410          1398  3                 END;
: 1411          1399  3
: 1412          1400  3             TES;
: 1413          1401  3         !
: 1414          1402  3         ! Signal error
: 1415          1403  3         !
: 1416          1404  3         loop_descr [0] = 2;
: 1417          1405  3         loop_descr [1] = mom$ab_service_data [svd$gk_pcno_lpc, svd$l_param];
: 1418          1406  3         mom_siglooperr (.code, .detail, loop_descr, .text, .flag);
: 1419          1407  3
: 1420          1408  2         END;
: 1421          1409  2
: 1422          1410  2 ctr = .iosb [ios$w_count] - 4;  ! Play games to look at the data
: 1423          1411  2 ctr = .ctr - CH$RCHAR (.ptr) - 1;         ! Skip over the device name
: 1424          1412  2 ptr = .ptr + CH$RCHAR (.ptr) + 1;         ! Set pointer to optional data
: 1425          1413  2
: 1426          1414  2 !
: 1427          1415  2 ! Verify optional data if Phase 3 mirror. (Ignore it if Phase 2.)
: 1428          1416  2 !
: 1429          1417  2 IF .version EQL mom$c_loop_phase3 THEN
: 1430          1418  3     BEGIN
: 1431          1419  4     IF NOT (CH$RCHAR (.ptr) EQL 2)       ! Verify optional data is 2 bytes long
: 1432          1420  3     OR .(.ptr + 1)<0,16,0> LSS
: 1433          1421  3         .mom$ab_service_data [svd$gk_pcno_lpl, svd$l_param] ! And mirror buffer size is greater
: 1434          1422  3                                                     ! than requested loop size
: 1435          1423  3     THEN
: 1436          1424  4         BEGIN
: 1437          1425  4         mirror_descr [0] = 2; ! Set up descriptor of mirror
: 1438          1426  4         mirror_descr [1] = .ptr + 1; ! Maximum buffer size
: 1439          1427  4         mom_siglooperr (nma$c_sts_pva,
: 1440          1428  4                         nma$c_pcno_lpl,
: 1441          1429  4                         mirror_descr,
: 1442          1430  4                         mom$_mirbfovf,
: 1443          1431  4                         msb$m_msg_fld); ! Signal error
: 1444          1432  3         END;
: 1445          1433  2     END;
: 1446          1434  2
: 1447          1435  2 RETURN
: 1448          1436  2
: 1449          1437  1 END;




                                                             .PSECT  $OWN$,NOEXE,2

                                    0020E            .BLKB   2
                          00000040  00210 MBX_MAXMSG:
                                                     .LONG   64                          ;
                          00000100  00214 MBX_BUFQUO:
                                                     .LONG   256                         ;
                                    00218 CHNCHAR:.BLKB   116
```

```
                                                        .PSECT  $CODE$,NOWRT,2

                                   03FC 00000 MOM_OPENLINK:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                       ; 1280
                 59 00000000V  00   9E 00002            MOVAB   MOM_SIGLOOPERR, R9                                 ;
                 58 00000000G  00   9E 00009            MOVAB   SYS$QIOW, R8                                       ;
                 57 00000000V  00   9E 00010            MOVAB   MOM_MAPLOOPERR, R7                                 ;
                 56 00000000'  00   9E 00017            MOVAB   LOOP_MBXCHAN, R6                                   ;
                 5E            34   C2 0001E            SUBL2   #52, SP                                            ;
                               56   DD 00021            PUSHL   R6                                                 ; 1318
                         FC    A6   9F 00023            PUSHAB  LOOP_CHAN                                          ;
                       0210    C6   9F 00026            PUSHAB  MBX_BUFQUO                                         ;
                       020C    C6   9F 0002A            PUSHAB  MBX_MAXMSG                                         ;
                  00000000G    00   9F 0002E            PUSHAB  MOM$GQ_NETNAMDSC                                   ;
       00000000G  00           05   FB 00034            CALLS   #5, LIB$ASN_WTH_MBX                                ;
                  53           50   D0 0003B            MOVL    R0, STATUS                                         ;
                  00000000*    00   DD 0003E            PUSHL   <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPC*137>-; 1328
                                                                >+9>                                               ;
                               7E   D4 00044            CLRL    -(SP)                                              ;
                               53   DD 00046            PUSHL   STATUS                                             ;
                 7E            15   CE 00048            MNEGL   #21, -(SP)                                         ;
                 67            04   FB 0004B            CALLS   #4, MOM_MAPLOOPERR                                 ;
                               7E   7C 0004E            CLRQ    -(SP)                                              ; 1336
                               7E   7C 00050            CLRQ    -(SP)                                              ;
                         1C    A6   9F 00052            PUSHAB  NFBDSC                                             ;
                               7E   7C 00055            CLRQ    -(SP)                                              ;
                               7E   D4 00057            CLRL    -(SP)                                              ;
                         30    AE   9F 00059            PUSHAB  IOSB                                               ;
                               32   DD 0005C            PUSHL   #50                                                ;
                         FC    A6   DD 0005E            PUSHL   LOOP_CHAN                                          ;
                               7E   D4 00061            CLRL    -(SP)                                              ;
                 68            0C   FB 00063            CALLS   #12, SYS$QIOW                                      ;
                 53            50   D0 00066            MOVL    R0, STATUS                                         ;
                  00000000*    00   DD 00069            PUSHL   <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPC*137>-; 1341
                                                                >+9>                                               ;
                         14    AE   9F 0006F            PUSHAB  IOSB                                               ; 1340
                               53   DD 00072            PUSHL   STATUS                                             ;
                 7E            15   CE 00074            MNEGL   #21, -(SP)                                         ;
                 67            04   FB 00077            CALLS   #4, MOM_MAPLOOPERR                                 ;
                               7E   7C 0007A            CLRQ    -(SP)                                              ; 1350
                               7E   7C 0007C            CLRQ    -(SP)                                              ;
                               28   DD 0007E            PUSHL   #40                                                ;
                         24    A6   9F 00080            PUSHAB  MBX_BUFFER                                         ;
                               7E   7C 00083            CLRQ    -(SP)                                              ;
                         30    AE   9F 00085            PUSHAB  IOSB                                               ;
                               31   DD 00088            PUSHL   #49                                                ;
                               66   DD 0008A            PUSHL   LOOP_MBXCHAN                                       ;
                               7E   D4 0008C            CLRL    -(SP)                                              ;
                 68            0C   FB 0008E            CALLS   #12, SYS$QIOW                                      ;
                 53            50   D0 00091            MOVL    R0, STATUS                                         ;
                  00000000*    00   DD 00094            PUSHL   <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPC*137>-; 1355
                                                                >+9>                                               ;
                         14    AE   9F 0009A            PUSHAB  IOSB                                               ; 1354
                               53   DD 0009D            PUSHL   STATUS                                             ;
                 7E            15   CE 0009F            MNEGL   #21, -(SP)                                         ;
```

```
                        67              04  FB 000A2          CALLS    #4, MOM_MAPLOOPERR                                     ;
                        53       24     A6  3C 000A5          MOVZWL   MBX_BUFFER, STATUS                                     ; 1359
                        52       28     A6  9E 000A9          MOVAB    MBX_BUFFER+4, PTR                                      ; 1360
                        31              53  D1 000AD          CMPL     STATUS, #49                                            ; 1362
                                        7A  13 000B0          BEQL     11$                                                    ;
                                        54  D4 000B2          CLRL     FLAG                                                   ; 1364
                        55              15  8E 000B4          MNEGB    #21, CODE                                              ; 1365
                                        51  D4 000B7          CLRL     TEXT                                                   ; 1366
                        30              53  D1 000B9          CMPL     STATUS, #48                                            ; 1370
                                        05  13 000BC          BEQL     1$                                                     ;
                        34              53  D1 000BE          CMPL     STATUS, #52                                            ;
                                        05  12 000C1          BNEQ     2$                                                     ;
                        50              0E  B0 000C3 1$:      MOVW     #14, DETAIL                                            ; 1372
                                        4A  11 000C6          BRB      10$                                                    ;
                        33              53  D1 000C8 2$:      CMPL     STATUS, #51                                            ; 1374
                                        05  12 000CB          BNEQ     3$                                                     ;
                        50              0D  B0 000CD          MOVW     #13, DETAIL                                            ; 1375
                                        40  11 000D0          BRB      10$                                                    ;
                        3B              53  D1 000D2 3$:      CMPL     STATUS, #59                                            ; 1377
                                        05  12 000D5          BNEQ     4$                                                     ;
                        50              0B  B0 000D7          MOVW     #11, DETAIL                                            ; 1378
                                        36  11 000DA          BRB      10$                                                    ;
                        36              53  D1 000DC 4$:      CMPL     STATUS, #54                                            ; 1380
                                        05  13 000DF          BEQL     5$                                                     ;
                        3D              53  D1 000E1          CMPL     STATUS, #61                                            ;
                                        05  12 000E4          BNEQ     6$                                                     ;
                        50              0C  B0 000E6 5$:      MOVW     #12, DETAIL                                            ; 1382
                                        27  11 000E9          BRB      10$                                                    ;
                        38              53  D1 000EB 6$:      CMPL     STATUS, #56                                            ; 1384
                                        05  12 000EE          BNEQ     7$                                                     ;
                        50              05  B0 000F0          MOVW     #5, DETAIL                                             ; 1385
                                        1D  11 000F3          BRB      10$                                                    ;
                        39              53  D1 000F5 7$:      CMPL     STATUS, #57                                            ; 1387
                                        05  12 000F8          BNEQ     8$                                                     ;
                        50              0F  B0 000FA          MOVW     #15, DETAIL                                            ; 1388
                                        13  11 000FD          BRB      10$                                                    ;
                        3A              53  D1 000FF 8$:      CMPL     STATUS, #58                                            ; 1390
                                        05  12 00102          BNEQ     9$                                                     ;
                        50              0A  B0 00104          MOVW     #10, DETAIL                                            ; 1391
                                        09  11 00107          BRB      10$                                                    ;
                        50              01  AE 00109 9$:      MNEGW    #1, DETAIL                                             ; 1395
                        54              04  88 0010C          BISB2    #4, FLAG                                               ; 1396
                        51              53  D0 0010F          MOVL     STATUS, TEXT                                           ; 1397
                   08   AE              02  D0 00112 10$:     MOVL     #2, LOOP_DESCR                                         ; 1404
                   0C   AE 00000000*    00  9E 00116          MOVAB    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPC*137>-           ; 1405
                                                                       >+9>, LOOP_DESCR+4                                     ;
                                        12  BB 0011E          PUSHR    #^M<R1,R4>                                             ; 1406
                                 10     AE  9F 00120          PUSHAB   LOOP_DESCR                                             ;
                        7E              50  3C 00123          MOVZWL   DETAIL, -(SP)                                          ;
                        7E              55  9A 00126          MOVZBL   CODE, -(SP)                                            ;
                        69              05  FB 00129          CALLS    #5, MOM_SIGLOOPERR                                     ;
                        50       12     AE  3C 0012C 11$:     MOVZWL   IOSB+2, CTR                                            ; 1410
                        50              04  C2 00130          SUBL2    #4, CTR                                                ;
                        51              62  9A 00133          MOVZBL   (PTR), R1                                              ; 1411
              53        50              51  C3 00136          SUBL3    R1, CTR, R3                                            ;
                        50       FF     A3  9E 0013A          MOVAB    -1(R3), CTR                                            ;
                        52       01 A142    9E 0013E          MOVAB    1(R1)[PTR], PTR                                        ; 1412
```

```
                                    01      0124   C6  D1 00143          CMPL     VERSION, #1                                  ; 1417
                                                   2E  12 00148          BNEQ     13$                                          ;
                                    02             62  91 0014A          CMPB     (PTR), #2                                    ; 1419
                                                   0C  12 0014D          BNEQ     12$                                          ;
00000000*  00        01    A2       10             00  ED 0014F          CMPZV    #0, #16, 1(PTR), <<MOM$AB_SERVICE_DATA+-     ; 1421
                                                                                  <SVD$GK_PCNO_LPL*137>>+9>                    ;
                                                   1D  18 00159          BGEQ     13$                                          ;
                                    6E             02  D0 0015B  12$:    MOVL     #2, MIRROR_DESCR                             ; 1425
                              04    AE        01   A2  9E 0015E          MOVAB    1(R2), MIRROR_DESCR+4                        ; 1426
                                                   04  DD 00163          PUSHL    #4                                           ; 1427
                                       00000000G   8F  DD 00165          PUSHL    #MOM$_MIRBFOVF                               ;
                                              08   AE  9F 0016B          PUSHAB   MIRROR_DESCR                                 ;
                                    7E        97   8F  9A 0016E          MOVZBL   #151, -(SP)                                  ;
                                    7E             10  CE 00172          MNEGL    #16, -(SP)                                   ;
                                    69             05  FB 00175          CALLS    #5, MOM_SIGLOOPERR                           ;
                                                       04 00178  13$:    RET                                                   ; 1437

; Routine Size:  377 bytes,    Routine Base:  $CODE$ + 08DF
; 1451              1438  1 %SBTTL 'mom_bldloopnfb  Build a Network Control Block'
; 1452              1439  1 ROUTINE mom_bldloopnfb : NOVALUE =
; 1453              1440  1
; 1454              1441  1 !++
; 1455              1442  1 ! FUNCTIONAL DESCRIPTION:
; 1456              1443  1 !
; 1457              1444  1 !       This routine builds the NFB for the loopback mirror
; 1458              1445  1 !       from information left around by the parse.
; 1459              1446  1 !
; 1460              1447  1 !--
; 1461              1448  1
; 1462              1449  2 BEGIN
; 1463              1450  2
; 1464              1451  2 MAP
; 1465              1452  2     mom$gb_option_byte: BBLOCK [1];
; 1466              1453  2
; 1467              1454  2 LOCAL
; 1468              1455  2     ptr,
; 1469              1456  2     ctr,
; 1470              1457  2     node_addr;
; 1471              1458  2
; 1472              1459  2 !
; 1473              1460  2 !       Initialize the NFB_BUFFER descriptor
; 1474              1461  2 !
; 1475              1462  2 nfbdsc [0] = nfb_bufsize;
; 1476              1463  2 nfbdsc [1] = nfb_buffer;
; 1477              1464  2 !
; 1478              1465  2 !       Obtain the node spec and strip trailing colons
; 1479              1466  2 !
; 1480              1467  2 IF .mom$gb_entity_code EQL mom$c_nodebyname THEN
; 1481              1468  2     ptr = CH$MOVE (.mom$gq_entity_buf_dsc [0],
; 1482              1469  2                 .mom$gq_entity_buf_dsc [1],
; 1483              1470  2                 .nfbdsc [1])
; 1484              1471  2 ELSE
; 1485              1472  3     BEGIN
; 1486              1473  3     node_addr = .(.mom$gq_entity_buf_dsc [1])<0,16>;
; 1487            P 1474  3     $FAOL (CTRSTR = $ASCID ('!UW'),
; 1488            P 1475  3            OUTLEN = ptr,
; 1489            P 1476  3            OUTBUF = nfbdsc,
; 1490              1477  3            PRMLST = node_addr);
; 1491              1478  3     ptr = .ptr<0,16> + .nfbdsc [1];
; 1492              1479  2     END;
; 1493              1480  2 !
; 1494              1481  2 !       Obtain the access control
; 1495              1482  2 !
; 1496              1483  2 IF .mom$gb_option_byte [nma$v_opt_acc] THEN       ! Is there access control in
; 1497              1484  3     BEGIN
; 1498              1485  3     IF .userdsc [0] NEQ 0 THEN              ! If not, use other access ctl
; 1499              1486  4         BEGIN
; 1500              1487  4         CH$WCHAR_A ('"', ptr);              ! Put it in standard form
; 1501              1488  4         ptr = CH$MOVE ( .userdsc [0],
; 1502              1489  4                         .userdsc [1],
; 1503              1490  4                         .ptr);
; 1504              1491  4
; 1505              1492  4         IF .passworddsc NEQ 0 THEN          ! A password??
; 1506              1493  5             BEGIN
; 1507              1494  5             CH$WCHAR_A (' ', ptr);
; 1508           1495  5              ptr = CH$MOVE (.passworddsc [0],
; 1509           1496  5                             .passworddsc [1],
; 1510           1497  5                             .ptr);
; 1511           1498  4              END;
; 1512           1499  4
; 1513           1500  4          IF .accountdsc NEQ 0 THEN        ! An account??
; 1514           1501  5              BEGIN
; 1515           1502  5              CH$WCHAR_A (' ', ptr);
; 1516           1503  5              ptr = CH$MOVE (      .accountdsc [0],
; 1517           1504  5                                   .accountdsc [1],
; 1518           1505  5                                   .ptr);
; 1519           1506  4              END;
; 1520           1507  4
; 1521           1508  4          CH$WCHAR_A ('"', ptr);  ! End the access control spec
; 1522           1509  3          END;
; 1523           1510  2      END;
; 1524           1511  2 !
; 1525           1512  2 !        Copy the object connect specification to the end
; 1526           1513  2 !
; 1527           1514  2 IF .version EQL mom$c_loop_phase3 THEN
; 1528           1515  2     ptr = CH$MOVE ( .object25dsc [0],
; 1529           1516  2                     .object25dsc [1],
; 1530           1517  2                     .ptr)
; 1531           1518  2 ELSE
; 1532           1519  2     ptr = CH$MOVE ( .object19dsc [0],
; 1533           1520  2                     .object19dsc [1],
; 1534           1521  2                     .ptr);
; 1535           1522  2
; 1536           1523  2 nfbdsc [0] = .ptr - .nfbdsc [1]; ! Save length of NFB
; 1537           1524  2
; 1538           1525  1 END;


                                                                 .PSECT  $PLIT$,NOWRT,NOEXE,2

                                 57  55  21   000D4 P.AAT:       .ASCII  \!UW\                                          ;
                                              000D7              .BLKB   1
                                  00000003    000D8 P.AAS:       .LONG   3                                              ;
                                  00000000'   000DC              .ADDRESS P.AAT                                         ;

                                                                 .EXTRN  SYS$FAOL

                                                                 .PSECT  $CODE$,NOWRT,2

                                        00FC 00000 MOM_BLDLOOPNFB:
                                                                 .WORD   Save R2,R3,R4,R5,R6,R7                         ; 1439
                    57 00000000'   00   9E 00002                 MOVAB   P.AAS, R7                                      ;
                    56 00000000'   00   9E 00009                 MOVAB   NFBDSC+4, R6                                   ;
                    5E             08   C2 00010                 SUBL2   #8, SP                                         ;
               FC   A6        6E   8F   9A 00013                 MOVZBL  #110, NFBDSC                                   ; 1462
                    66        2C   A6   9E 00018                 MOVAB   NFB_BUFFER, NFBDSC+4                           ; 1463
                    51 00000000G   00   D0 0001C                 MOVL    MOM$GQ_ENTITY_BUF_DSC+4, R1                    ; 1469
                    01 00000000G   00   91 00023                 CMPB    MOM$GB_ENTITY_CODE, #1                         ; 1467
                              11   12   0002A                    BNEQ    1$                                             ;
                    50             66   D0 0002C                 MOVL    NFBDSC+4, R0                                   ; 1470
        60          61 00000000G   00   28 0002F                 MOVC3   MOM$GQ_ENTITY_BUF_DSC, (R1), (R0)              ;
```

```
                04  AE               53 D0 00037      MOVL    R3, PTR                         ;
                                     1D 11 0003B      BRB     2$                              ; 1468
                    6E               61 3C 0003D 1$:  MOVZWL  (R1), NODE_ADDR                 ; 1473
                                     5E DD 00040      PUSHL   SP                              ; 1477
                                FC   A6 9F 00042      PUSHAB  NFBDSC                          ;
                                0C   AE 9F 00045      PUSHAB  PTR                             ;
                                     57 DD 00048      PUSHL   R7                              ;
         00000000G  00               04 FB 0004A      CALLS   #4, SYS$FAOL                    ;
                    50          04   AE 3C 00051      MOVZWL  PTR, R0                         ; 1478
04   AE             50               66 C1 00055      ADDL3   NFBDSC+4, R0, PTR               ;
                         00000000G   00 95 0005A 2$:  TSTB    MOM$GB_OPTION_BYTE              ; 1483
                                     55 18 00060      BGEQ    5$                              ;
                    51          E4   A6 D0 00062      MOVL    USERDSC, R1                     ; 1485
                                     4F 13 00066      BEQL    5$                              ;
                04  BE               22 90 00068      MOVB    #34, @PTR                       ; 1487
                                04   AE D6 0006C      INCL    PTR                             ;
                    50          E8   A6 D0 0006F      MOVL    USERDSC+4, R0                   ; 1489
04   BE             60               51 28 00073      MOVC3   R1, (R0), @PTR                  ; 1490
                04  AE               53 D0 00078      MOVL    R3, PTR                         ;
                    51          F4   A6 D0 0007C      MOVL    PASSWORDDSC, R1                 ; 1492
                                     14 13 00080      BEQL    3$                              ;
                04  BE               20 90 00082      MOVB    #32, @PTR                       ; 1494
                                04   AE D6 00086      INCL    PTR                             ;
                    50          F8   A6 D0 00089      MOVL    PASSWORDDSC+4, R0               ; 1496
04   BE             60               51 28 0008D      MOVC3   R1, (R0), @PTR                  ; 1497
                04  AE               53 D0 00092      MOVL    R3, PTR                         ;
                    51          EC   A6 D0 00096 3$:  MOVL    ACCOUNTDSC, R1                  ; 1500
                                     14 13 0009A      BEQL    4$                              ;
                04  BE               20 90 0009C      MOVB    #32, @PTR                       ; 1502
                                04   AE D6 000A0      INCL    PTR                             ;
                    50          F0   A6 D0 000A3      MOVL    ACCOUNTDSC+4, R0                ; 1504
04   BE             60               51 28 000A7      MOVC3   R1, (R0), @PTR                  ; 1505
                04  AE               53 D0 000AC      MOVL    R3, PTR                         ;
                04  BE               22 90 000B0 4$:  MOVB    #34, @PTR                       ; 1508
                                04   AE D6 000B4      INCL    PTR                             ;
                    01        0104   C6 D1 000B7 5$:  CMPL    VERSION, #1                     ; 1514
                                     0E 12 000BC      BNEQ    6$                              ;
                    50        FF58   C7 D0 000BE      MOVL    OBJECT25DSC+4, R0               ; 1516
04   BE             60        FF54   C7 28 000C3      MOVC3   OBJECT25DSC, (R0), @PTR         ; 1517
                                     0C 11 000CA      BRB     7$                              ;
                    50        FF7C   C7 D0 000CC 6$:  MOVL    OBJECT19DSC+4, R0               ; 1520
04   BE             60        FF78   C7 28 000D1      MOVC3   OBJECT19DSC, (R0), @PTR         ; 1521
                04  AE               53 D0 000D8 7$:  MOVL    R3, PTR                         ;
FC   A6         04  AE               66 C3 000DC      SUBL3   NFBDSC+4, PTR, NFBDSC           ; 1523
                                        04 000E2      RET                                     ; 1525

; Routine Size:  227 bytes,    Routine Base:  $CODE$ + 0A58
```

```
1 %SBTTL 'mom_maplooperr      Map and signal MOM error'
1 ROUTINE mom_maplooperr (code, status, iosb, loop) : NOVALUE=
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 ! This routine takes system service status and QIO IOSB status
1 ! return codes and converts them to the proper detail codes for
1 ! the network management status codes of NMA$C_STS_MLD and NMA$C_STS_MLD.
1 !
1 ! The routine will also change the status to reflect the appropiate status
1 ! if the system service or QIO IOSB status maps such a network management
1 ! status code.(NMA$C_STS_PRI,NMA$C_STS_RES)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !     code        The network management status code to use
1 !     status      The system service returned status code to map
1 !     iosb        The address of the IOSB,if there  is one
1 !                 the IOS$W_STATUS status field is mapped if possible
1 !     loop        The number of times yet to loop
1 !
1 !--
2 BEGIN
2
2 MAP
2     iosb   : REF $iosb,
2     code   : BYTE,
2     status : WORD,
2     loop   : WORD;
2
2 LOCAL
2     detail      : WORD,                     ! Detail word for MOM_SIGLOOPERR
2     text,                                   ! Error text for MOM_SIGLOOPERR
2     flag,                                   ! Optional flag for MOM_SIGLOOPERR
2     count_descr : VECTOR [2];               ! Descriptor for LOOP
2
2 !
2 ! Check system status first if SS$_NORMAL check IOSB status if there is one.
2 ! If status is normal return from routine.  Otherwise attempt to map status
2 ! (either system service or IOSB status which ever fails first) into a
2 ! network management error detail or status code.
2 !
2 IF .status THEN
2     IF .iosb NEQA 0 THEN status = .iosb [ios$w_status];
2
2 IF .status THEN
2     RETURN success;
2 !
2 ! At this point the system service or the I/O has failed and STATUS
2 ! contains the system error code.
2 !
2 ! Attempt to map into a network error detail code.
2 !
2 text = 0;                                   ! Assume no optional text
2 flag = 0;                                   ! Assume no optional flags
2
```

```
2 SELECTONE .status OF
2     SET
2     [ss$_abort]:
2         detail = nma$c_ncedtl_die;          ! Node or object failed
2
2     [ss$_nosuchnode]:
2         detail = nma$c_ncedtl_una;          ! Unrecognized node name
2
2     [ss$_unreachable]:
2         BEGIN
3         detail = nma$c_ncedtl_unr;          ! Node unreachable
3         flag = msb$m_msg_fld;               ! Map TEXT to system error
3         text = .status;                     ! Pass system error code
2         END;
2
2     [ss$_invlogin]:
2         detail = nma$c_ncedtl_acc;          ! Access control rejected
2
2     [ss$_nolinks,
2      ss$_remrsrc]:
2         BEGIN
3         detail = nma$c_ncedtl_rsc;          ! Network resources
3         flag = msb$m_msg_fld;               ! Map TEXT to system error
3         text = .status;                     ! Pass system error code
2         END;
2
2     [ss$_nosuchobj]:
2         detail = nma$c_ncedtl_obj;          ! Unrecognized object
2
2     [ss$_thirdparty]:
2         detail = nma$c_ncedtl_abm;          ! Abort by management
2
2     [ss$_timeout]:
2         detail = nma$c_ncedtl_nrs;          ! No response from object
2
2     [ss$_ivdevnam]:
2         BEGIN
3         detail = nma$c_ncedtl_obj;          ! Assume no such object
3         text = mom$_ncbfail;                ! Say error in NCB format
2         END;
2
2     [ss$_reject]:
2         BEGIN
3         !
3         ! SS$_REJECT "real" reason for reject is in second longword of IOSB
3         ! Therefore we must map each possible reason
3         !
3         SELECTONE .iosb [4,0,16,0] OF
3             SET
3             [net$c_dr_access,
3              net$c_dr_accnt]:
3                 detail = nma$c_ncedtl_acc; ! Access control rejected
3
3             [net$c_dr_exit]:
3                 detail = nma$c_ncedtl_die; ! Node or object failed
3
3             [net$c_dr_nobj]:
```

```
; 1654      1640  3                detail = nma$c_ncedtl_obj; ! Unrecognized object
; 1655      1641  3
; 1656      1642  3            [net$c_dr_nocon]:
; 1657      1643  3                detail = nma$c_ncedtl_bsy; ! Object too busy
; 1658      1644  3
; 1659      1645  3            [net$c_dr_nopath]:
; 1660      1646  3                detail = nma$c_ncedtl_una; ! Unrecognized node name
; 1661      1647  3
; 1662      1648  3            [net$c_dr_normal]:
; 1663      1649  3                detail = nma$c_ncedtl_rjc; ! Rejected by object
; 1664      1650  3
; 1665      1651  3            [net$c_dr_segsiz,
; 1666      1652  3             net$c_dr_rsu]:
; 1667      1653  3                detail = nma$c_ncedtl_rsc; ! Network resource
; 1668      1654  3
; 1669      1655  3            [net$c_dr_shut]:
; 1670      1656  3                detail = nma$c_ncedtl_nsd; ! Node shutdown
; 1671      1657  3
; 1672      1658  3            [net$c_dr_third]:
; 1673      1659  3                detail = nma$c_ncedtl_abm; ! Abort by management
; 1674      1660  3
; 1675      1661  3            [OTHERWISE]:
; 1676      1662  4                BEGIN
; 1677      1663  4                detail = -1;            ! No detail
; 1678      1664  4                flag = .flag or msb$m_msg_fld;
; 1679      1665  4                text = .status;
; 1680      1666  3                END;
; 1681      1667  3
; 1682      1668  3            TES;
; 1683      1669  2        END;
; 1684      1670  2
; 1685      1671  2    [OTHERWISE]:
; 1686      1672  3        BEGIN
; 1687      1673  3        detail = -1;                   ! No detail
; 1688      1674  3        flag = .flag OR msb$m_msg_fld;  ! Map TEXT to system error
; 1689      1675  3        text = .status;                ! Pass system error code
; 1690      1676  2        END;
; 1691      1677  2
; 1692      1678  2    TES;
; 1693      1679  2  !
; 1694      1680  2  ! Build the data descriptor of number of messages not looped for MOM_SIGLOOPERR
; 1695      1681  2  !
; 1696      1682  2 count_descr [0] = 2;
; 1697      1683  2 count_descr [1] = loop;
; 1698      1684  2  !
; 1699      1685  2  ! Signal error
; 1700      1686  2  !
; 1701      1687  2 mom_siglooperr (.code, .detail, count_descr, .text, .flag);
; 1702      1688  2
; 1703      1689  1 END;                            ! End of MOM_MAPLOOPERR



                                  001C 00000 MOM_MAPLOOPERR:
                                                       .WORD   Save R2,R3,R4                         ; 1527
```

```
          5E               08 C2 00002         SUBL2   #8, SP                  ;
          0A         08    AC E9 00005         BLBC    STATUS, 1$              ; 1569
                     0C    AC D5 00009         TSTL    IOSB                    ; 1570
                           05 13 0000C         BEQL    1$                      ;
      08  AC         0C    BC B0 0000E         MOVW    @IOSB, STATUS           ;
          52         08    AC 3C 00013 1$:     MOVZWL  STATUS, R2              ; 1572
          01               52 E9 00017         BLBC    R2, 2$                  ;
                              04 0001A         RET                             ;
                           53 7C 0001B 2$:     CLRQ    TEXT                    ; 1580
          2C               52 B1 0001D         CMPW    R2, #44                 ; 1585
                           03 12 00020         BNEQ    3$                      ;
                         0089 31 00022         BRW     14$                     ;
    028C  8F               52 B1 00025 3$:     CMPW    R2, #652                ; 1588
                           03 12 0002A         BNEQ    4$                      ;
                         009D 31 0002C         BRW     19$                     ;
    2094  8F               52 B1 0002F 4$:     CMPW    R2, #8340               ; 1591
                           05 12 00034         BNEQ    5$                      ;
          51               03 B0 00036         MOVW    #3, DETAIL              ; 1593
                           18 11 00039         BRB     7$                      ; 1594
    209C  8F               52 B1 0003B 5$:     CMPW    R2, #8348               ; 1598
                           62 13 00040         BEQL    12$                     ;
    027C  8F               52 B1 00042         CMPW    R2, #636                ; 1601
                           07 13 00047         BEQL    6$                      ;
    206C  8F               52 B1 00049         CMPW    R2, #8300               ;
                           09 12 0004E         BNEQ    8$                      ;
          51               04 B0 00050 6$:     MOVW    #4, DETAIL              ; 1604
          54               04 D0 00053 7$:     MOVL    #4, FLAG                ; 1605
                         00AA 31 00056         BRW     28$                     ; 1606
    20A4  8F               52 B1 00059 8$:     CMPW    R2, #8356               ; 1609
                           58 13 0005E         BEQL    16$                     ;
    207C  8F               52 B1 00060         CMPW    R2, #8316               ; 1612
                           03 12 00065         BNEQ    9$                      ;
                         008E 31 00067         BRW     26$                     ;
    022C  8F               52 B1 0006A 9$:     CMPW    R2, #556                ; 1615
                           05 12 0006F         BNEQ    10$                     ;
          51               0A B0 00071         MOVW    #10, DETAIL             ; 1616
                           7B 11 00074         BRB     24$                     ;
    0144  8F               52 B1 00076 10$:    CMPW    R2, #324                ; 1618
                           0C 12 0007B         BNEQ    11$                     ;
          51               07 B0 0007D         MOVW    #7, DETAIL              ; 1620
          53  00000000G    8F D0 00080         MOVL    #MOM$_NCBFAIL, TEXT     ; 1621
                           7D 11 00087         BRB     29$                     ; 1583
    0294  8F               52 B1 00089 11$:    CMPW    R2, #660                ; 1624
                           6D 12 0008E         BNEQ    27$                     ;
          50         0C    AC D0 00090         MOVL    IOSB, R0                ; 1630
          50               04 C0 00094         ADDL2   #4, R0                  ;
          50               60 3C 00097         MOVZWL  (R0), R0                ;
          22               50 B1 0009A         CMPW    R0, #34                 ; 1632
                           05 13 0009D         BEQL    12$                     ;
          24               50 B1 0009F         CMPW    R0, #36                 ;
                           05 12 000A2         BNEQ    13$                     ;
          51               08 B0 000A4 12$:    MOVW    #8, DETAIL              ; 1634
                           5D 11 000A7         BRB     29$                     ;
          26               50 B1 000A9 13$:    CMPW    R0, #38                 ; 1636
                           05 12 000AC         BNEQ    15$                     ;
          51               0C B0 000AE 14$:    MOVW    #12, DETAIL             ; 1637
                           53 11 000B1         BRB     29$                     ;
```

```
                          04              50  B1 000B3 15$:      CMPW     R0, #4                                       ; 1639
                                          05  12 000B6           BNEQ     17$                                          ;
                          51              07  B0 000B8 16$:      MOVW     #7, DETAIL                                   ; 1640
                                          49  11 000BB           BRB      29$                                          ;
                          20              50  B1 000BD 17$:      CMPW     R0, #32                                      ; 1642
                                          05  12 000C0           BNEQ     18$                                          ;
                          51              09  B0 000C2           MOVW     #9, DETAIL                                   ; 1643
                                          3F  11 000C5           BRB      29$                                          ;
                          27              50  B1 000C7 18$:      CMPW     R0, #39                                      ; 1645
                                          05  12 000CA           BNEQ     20$                                          ;
                          51              02  B0 000CC 19$:      MOVW     #2, DETAIL                                   ; 1646
                                          35  11 000CF           BRB      29$                                          ;
                                          50  D5 000D1 20$:      TSTL     R0                                           ; 1648
                                          05  12 000D3           BNEQ     21$                                          ;
                          51              05  B0 000D5           MOVW     #5, DETAIL                                   ; 1649
                                          2C  11 000D8           BRB      29$                                          ;
                          01              50  B1 000DA 21$:      CMPW     R0, #1                                       ; 1651
                                          05  13 000DD           BEQL     22$                                          ;
                          25              50  B1 000DF           CMPW     R0, #37                                      ;
                                          05  12 000E2           BNEQ     23$                                          ;
                          51              04  B0 000E4 22$:      MOVW     #4, DETAIL                                   ; 1653
                                          1D  11 000E7           BRB      29$                                          ;
                          03              50  B1 000E9 23$:      CMPW     R0, #3                                       ; 1655
                                          05  12 000[illegible]C           BNEQ     25$                                          ;
                          51              0B  B0 000EE           MOVW     #11, DETAIL                                  ; 1656
                                          13  11 000F1 24$:      BRB      29$                                          ;
                          08              50  B1 000F3 25$:      CMPW     R0, #8                                       ; 1658
                                          05  12 000F6           BNEQ     27$                                          ;
                          51              0F  B0 000F8 26$:      MOVW     #15, DETAIL                                  ; 1659
                                          09  11 000FB           BRB      29$                                          ;
                          51              01  AE 000FD 27$:      MNEGW    #1, DETAIL                                   ; 1673
                          54              04  88 00100           BISB2    #4, FLAG                                     ; 1674
                          53              52  D0 00103 28$:      MOVL     R2, TEXT                                     ; 1675
                          6E              02  D0 00106 29$:      MOVL     #2, COUNT DESCR                              ; 1682
                   04     AE      10      AC  9E 00109           MOVAB    LOOP, COUNT_DESCR+4                          ; 1683
                                          18  BB 0010E           PUSHR    #^M<R3,R4>                                   ; 1687
                                  08      AE  9F 00110           PUSHAB   COUNT DESCR                                  ;
                          7E              51  3C 00113           MOVZWL   DETAIL, -(SP)                                ;
                          7E      04      AC  9A 00116           MOVZBL   CODE, -(SP)                                  ;
               00000000V  00              05  FB 0011A           CALLS    #5, MOM_SIGLOOPERR                           ;
                                              04 00121           RET                                                   ; 1689

; Routine Size:  290 bytes,    Routine Base:  $CODE$ + 0B3B
; 1705     1690  1 %SBTTL 'mom_siglooperr        Signal looper errors'
; 1706     1691  1 ROUTINE mom_siglooperr (code, detail, data, text, flag) : NOVALUE =
; 1707     1692  1
; 1708     1693  1 !++
; 1709     1694  1 ! FUNCTIONAL DESCRIPTION:
; 1710     1695  1 !
; 1711     1696  1 !       This routine builds the message segment block and actually signals
; 1712     1697  1 !       the error.
; 1713     1698  1 !
; 1714     1699  1 ! FORMAL PARAMETERS:
; 1715     1700  1 !
; 1716     1701  1 !       CODE            Code of the status to be signalled.
; 1717     1702  1 !       DETAIL          Detail to be signalled.
; 1718     1703  1 !       DATA            Address of the data descriptor.
; 1719     1704  1 !       TEXT            Code for optional detail error text.
; 1720     1705  1 !       FLAG            Mask for optional flags (used for MSB$M_MSG_FLD).
; 1721     1706  1 !
; 1722     1707  1 ! IMPLICIT OUTPUTS:
; 1723     1708  1 !
; 1724     1709  1 !       This routine will build the message segment block in MOM$AB_MSGBLOCK
; 1725     1710  1 !       This routine calls MOM$BLD_REPLY which builds the message in
; 1726     1711  1 !       MOM$AB_NICE_XMIT_BUF.
; 1727     1712  1 !
; 1728     1713  1 !--
; 1729     1714  2 BEGIN
; 1730     1715  2
; 1731     1716  2 MAP
; 1732     1717  2     code   : BYTE,
; 1733     1718  2     detail : WORD,
; 1734     1719  2     data   : REF VECTOR,
; 1735     1720  2     text   : LONG,
; 1736     1721  2     flag   : LONG;
; 1737     1722  2
; 1738     1723  2 LOCAL
; 1739     1724  2     message_size;
; 1740     1725  2
; 1741     1726  2 !
; 1742     1727  2 ! Build the message descriptor block in MOM$AB_MSGBLOCK
; 1743     1728  2 !
; 1744     1729  2 mom$ab_msgblock [msb$l_flags] = .flag OR
; 1745     1730  2                                  msb$m_det_fld OR
; 1746     1731  2                                  msb$m_data_fld;
; 1747     1732  2 mom$ab_msgblock [msb$b_code] = .code;     ! Status code
; 1748     1733  2 mom$ab_msgblock [msb$w_detail] = .detail; ! Detail code
; 1749     1734  2 mom$ab_msgblock [msb$a_data] = .data;     ! Count field
; 1750     1735  2 mom$ab_msgblock [msb$l_text] = .text;     ! Optional error text code
; 1751     1736  2 !
; 1752     1737  2 ! Call MOM$BLD_REPLY to build the error message
; 1753     1738  2 !
; 1754     1739  2 mom$bld_reply (mom$ab_msgblock, message_size);
; 1755     1740  2 !
; 1756     1741  2 ! Signal the message
; 1757     1742  2 !
; 1758     1743  2
; 1759     1744  2 $signal_msg (mom$ab_nice_xmit_buf, .message_size);
; 1760     1745  2
; 1761     1746  1 END;                                ! End of mom_siglooperr
```

```
                                                   0004 00000 MOM_SIGLOOPERR:
                                                                    .WORD   Save R2                                  ; 1691
                                    52 00000000G 00  9E 00002       MOVAB   MOM$AB_MSGBLOCK, R2                      ;
                                    5E           04  C2 00009       SUBL2   #4, SP                                   ;
                       62      14   AC           22  C9 0000C       BISL3   #34, FLAG, MOM$AB_MSGBLOCK               ; 1730
                               04   A2        04 AC  90 00011       MOVB    CODE, MOM$AB_MSGBLOCK+4                  ; 1732
                               08   A2        08 AC  B0 00016       MOVW    DETAIL, MOM$AB_MSGBLOCK+8                ; 1733
                               18   A2        0C AC  D0 0001B       MOVL    DATA, MOM$AB_MSGBLOCK+24                 ; 1734
                               0C   A2        10 AC  D0 00020       MOVL    TEXT, MOM$AB_MSGBLOCK+12                 ; 1735
                                           4004  8F  BB 00025       PUSHR   #^M<R2,SP>                               ; 1739
                          00000000G 00           02  FB 00029       CALLS   #2, MOM$BLD_REPLY                        ;
                                                 6E  DD 00030       PUSHL   MESSAGE_SIZE                             ; 1744
                                       00000000G 00  9F 00032       PUSHAB  MOM$AB_NICE_XMIT_BUF                     ;
                                       02070000  8F  DD 00038       PUSHL   #34013184                                ;
                          00000000G 00           03  FB 0003E       CALLS   #3, LIB$SIGNAL                           ;
                                                     04 00045       RET                                              ; 1746

; Routine Size:  70 bytes,    Routine Base:  $CODE$ + 0C5D
; 1763          1747  1 %SBTTL 'mom$saveuser      NPARSE action routine'
; 1764          1748  1 GLOBAL ROUTINE  mom$saveuser    =
; 1765          1749  1
; 1766          1750  1 !++
; 1767          1751  1 ! FUNCTIONAL DESCRIPTION:
; 1768          1752  1 !
; 1769          1753  1 ! This routine builds the descriptor for the USER part of
; 1770          1754  1 ! the access control string
; 1771          1755  1 !
; 1772          1756  1 ! IMPLICIT INPUTS:
; 1773          1757  1 !       NPARSE block is pointed by AP and defined in NPARSE macro
; 1774          1758  1 !
; 1775          1759  1 ! ROUTINE VALUE:
; 1776          1760  1 ! COMPLETION CODE:
; 1777          1761  1 !       SUCCESS always returned
; 1778          1762  1 !
; 1779          1763  1 ! SIDE EFFECTS:
; 1780          1764  1 !       A descriptor is built in USERDSC
; 1781          1765  1 !--
; 1782          1766  2 BEGIN
; 1783          1767  2
; 1784          1768  2 $npa_argdef;
; 1785          1769  2
; 1786          1770  2 userdsc [0] = .nparse_block [npa$l_fldcnt] - 1;
; 1787          1771  2 userdsc [1] = .nparse_block [npa$l_fldptr] + 1;
; 1788          1772  2
; 1789          1773  2 RETURN success
; 1790          1774  2
; 1791          1775  1 END;                               ! End of mom$saveuser



                                                  0000 00000           .ENTRY  MOM$SAVEUSER, Save nothing                     ; 1748
           00000000'  00        10   AC        01   C3 00002           SUBL3   #1, 16(NPARSE_BLOCK), USERDSC                  ; 1770
           00000000'  00        14   AC        01   C1 0000B           ADDL3   #1, 20(NPARSE_BLOCK), USERDSC+4                ; 1771
                                     50        01   D0 00014           MOVL    #1, R0                                         ; 1773
                                                    04 00017           RET                                                    ; 1775

; Routine Size:  24 bytes,    Routine Base:  $CODE$ + 0CA3
; 1793          1776  1 %SBTTL 'mom$savepasswrd        NPARSE action routine'
; 1794          1777  1 GLOBAL ROUTINE mom$savepasswrd =
; 1795          1778  1
; 1796          1779  1 !++
; 1797          1780  1 ! FUNCTIONAL DESCRIPTION:
; 1798          1781  1 !
; 1799          1782  1 ! This routine creates a descriptor for the password portion
; 1800          1783  1 ! of the access control string and stores it in PASSWORDDSC
; 1801          1784  1 !
; 1802          1785  1 ! IMPLICIT INPUTS:
; 1803          1786  1 !         NPARSE block is pointed to by AP and is defined in $NPA_ARGDEF macro
; 1804          1787  1 !
; 1805          1788  1 ! COMPLETION CODE:
; 1806          1789  1 !         SUCCESS returned
; 1807          1790  1 !
; 1808          1791  1 ! SIDE EFFECTS:
; 1809          1792  1 !         Descriptor built in PASSWORDDSC
; 1810          1793  1 !
; 1811          1794  1 !--
; 1812          1795  2 BEGIN
; 1813          1796  2
; 1814          1797  2 $npa_argdef;
; 1815          1798  2
; 1816          1799  2 passworddsc [0] = .nparse_block [npa$l_fldcnt] - 1;
; 1817          1800  2 passworddsc [1] = .nparse_block [npa$l_fldptr] + 1;
; 1818          1801  2
; 1819          1802  2 RETURN success
; 1820          1803  1 END;                                      ! End of MOM$SAVEPASSWRD



                                                 0000 00000     .ENTRY  MOM$SAVEPASSWRD, Save nothing                   ; 1777
        00000000'  00       10   AC          01   C3 00002     SUBL3   #1, 16(NPARSE_BLOCK), PASSWORDDSC               ; 1799
        00000000'  00       14   AC          01   C1 0000B     ADDL3   #1, 20(NPARSE_BLOCK), PASSWORDDSC+4             ; 1800
                                 50          01   D0 00014     MOVL    #1, R0                                          ; 1802
                                                  04 00017     RET                                                     ; 1803

; Routine Size:  24 bytes,    Routine Base:  $CODE$ + 0CBB
; 1822          1804  1 %SBTTL 'mom$saveacct  NPARSE action routine'
; 1823          1805  1 GLOBAL ROUTINE mom$saveacct =
; 1824          1806  1
; 1825          1807  1 !++
; 1826          1808  1 ! FUNCTIONAL DESCRIPTION:
; 1827          1809  1 !
; 1828          1810  1 ! This routine builds a descriptor for the account portion of
; 1829          1811  1 ! the access control string.
; 1830          1812  1 !
; 1831          1813  1 ! IMPLICIT INPUTS:
; 1832          1814  1 !       NPARSE block is pointed by AP and defined in $NPA_ARGDEF macro
; 1833          1815  1 !
; 1834          1816  1 ! ROUTINE VALUE:
; 1835          1817  1 ! COMPLETION CODE:
; 1836          1818  1 !       SUCCESS returned
; 1837          1819  1 !
; 1838          1820  1 ! SIDE EFFECTS:
; 1839          1821  1 !       A descriptor is built in ACCOUNTDSC
; 1840          1822  1 !--
; 1841          1823  2 BEGIN
; 1842          1824  2
; 1843          1825  2 $npa_argdef;
; 1844          1826  2
; 1845          1827  2 accountdsc [0] = .nparse_block [npa$l_fldcnt] - 1;
; 1846          1828  2 accountdsc [1] = .nparse_block [npa$l_fldptr] + 1;
; 1847          1829  2
; 1848          1830  2 RETURN success
; 1849          1831  2
; 1850          1832  1 END;                                    ! End of MOM$SAVEACCT



                                                      0000 00000          .ENTRY  MOM$SAVEACCT, Save nothing                           ; 1805
          00000000'  00       10   AC             01   C3 00002           SUBL3   #1, 16(NPARSE_BLOCK), ACCOUNTDSC                     ; 1827
          00000000'  00       14   AC             01   C1 0000B           ADDL3   #1, 20(NPARSE_BLOCK), ACCOUNTDSC+4                   ; 1828
                                   50             01   D0 00014           MOVL    #1, R0                                               ; 1830
                                                       04 00017           RET                                                          ; 1832

; Routine Size:  24 bytes,     Routine Base:  $CODE$ + 0CD3
; 1852          1833  1 %SBTTL 'mom$loophandler  Condition handler'
; 1853          1834  1 GLOBAL ROUTINE mom$loophandler (signal_vec, mechanism)  =
; 1854          1835  1
; 1855          1836  1 !++
; 1856          1837  1 ! FUNCTIONAL DESCRIPTION:
; 1857          1838  1 !
; 1858          1839  1 !       This routine is a condition handler that performs cleanup
; 1859          1840  1 !       at the end of loop operations.  Any buffers that were
; 1860          1841  1 !       allocated from virtual memory are deallocated.
; 1861          1842  1 !
; 1862          1843  1 ! FORMAL PARAMETERS:
; 1863          1844  1 !
; 1864          1845  1 !       SIGNAL_VEC      Pointer to the signal vector.
; 1865          1846  1 !       MECHANISM       Pointer to the mechanism array.
; 1866          1847  1 !
; 1867          1848  1 !--
; 1868          1849  2 BEGIN
; 1869          1850  2
; 1870          1851  2 MAP
; 1871          1852  2     signal_vec : REF BBLOCK,       ! Signal vector argument
; 1872          1853  2     mechanism  : REF BBLOCK;       ! Mechanism vector array pointer
; 1873          1854  2
; 1874          1855  2 !
; 1875          1856  2 ! If loop buffers were allocated, deallocate them.
; 1876          1857  2 !
; 1877          1858  2 mom$freebuffer ();
; 1878          1859  2 !
; 1879          1860  2 ! If it was a LOOP with ASSIST command, deassign the assist channel here.
; 1880          1861  2 ! The target channel will be deassigned in either MOM$AUTOHANDLER or
; 1881          1862  2 ! MOM$SERVICEHANDLER.
; 1882          1863  2 !
; 1883          1864  2 IF .mom$ab_loop_cib [cib$l_chan] NEQ 0 THEN
; 1884          1865  2     $DASSGN (CHAN = .mom$ab_loop_cib [cib$l_chan]);
; 1885          1866  2
; 1886          1867  2 RETURN ss$_resignal;              ! Always resignal error
; 1887          1868  2
; 1888          1869  1 END;                              ! End of MOM$LOOPHANDLER





                                       0000 00000          .ENTRY  MOM$LOOPHANDLER, Save nothing                        ; 1834
                     FBD1   CF           00  FB 00002          CALLS   #0, MOM$FREEBUFFER                                   ; 1858
                            50 00000000G 00  D0 00007          MOVL    MOM$AB_LOOP_CIB, R0                                  ; 1864
                                         09  13 0000E          BEQL    1$                                                   ;
                                         50  DD 00010          PUSHL   R0                                                   ; 1865
                00000000G   00           01  FB 00012          CALLS   #1, SYS$DASSGN                                       ;
                            50     0918  8F  3C 00019 1$:      MOVZWL  #2328, R0                                            ; 1867
                                             04 0001E          RET                                                          ; 1869

; Routine Size:  31 bytes,    Routine Base:  $CODE$ + 0CEB
; 1890          1870  1 %SBTTL 'mom$testhandler  Condition handler'
; 1891          1871  1 GLOBAL ROUTINE  mom$testhandler (signal_vec, mechanism) =
; 1892          1872  1
; 1893          1873  1 !++
; 1894          1874  1 ! FUNCTIONAL DESCRIPTION:
; 1895          1875  1 !       This routine is the condition handler to force a disconnect of
; 1896          1876  1 !       the mirror link on any errors
; 1897          1877  1 !
; 1898          1878  1 !       This routine also deallocates any buffers allocated.
; 1899          1879  1 !
; 1900          1880  1 ! FORMAL PARAMETERS:
; 1901          1881  1 !       SIGNAL_VEC      Pointer to the signal vector.
; 1902          1882  1 !       MECHANISM       Pointer to then mechanism array.
; 1903          1883  1 !
; 1904          1884  1 ! IMPLICIT INPUTS:
; 1905          1885  1 !       LOOP_CHAN       The channel that the connect to the mirror
; 1906          1886  1 !                       is assumed to have been used.
; 1907          1887  1 !
; 1908          1888  1 !--
; 1909          1889  2 BEGIN
; 1910          1890  2
; 1911          1891  2 MAP
; 1912          1892  2     signal_vec : REF BBLOCK,            ! Signal vector argument
; 1913          1893  2     mechanism  : REF BBLOCK;            ! Mechanism vector array pointer
; 1914          1894  2
; 1915          1895  2 LOCAL
; 1916          1896  2     status_code : BBLOCK [4];           ! Status code
; 1917          1897  2
; 1918          1898  2 status_code = .signal_vec [chf$l_sig_name]; ! Save condition name
; 1919          1899  2
; 1920          1900  2 IF .status_code [sts$v_fac_no] EQLU mom$k_fac_code THEN
; 1921          1901  3     BEGIN
; 1922          1902  3     !
; 1923          1903  3     ! Deassign the mirror channel.
; 1924          1904  3     !
; 1925          1905  3     $DASSGN (CHAN = .loop_chan);
; 1926          1906  3     !
; 1927          1907  3     ! If loop buffers were allocated, deallocate them.
; 1928          1908  3     !
; 1929          1909  3     mom$freebuffer ();
; 1930          1910  3
; 1931          1911  2     END;
; 1932          1912  2
; 1933          1913  2 RETURN ss$_resignal;                 ! Always resignal error
; 1934          1914  2
; 1935          1915  1 END;                                 ! End of MOM$TESTHANDLER



                                             0000 00000          .ENTRY  MOM$TESTHANDLER, Save nothing                 ; 1871
                           50      04     AC   D0 00002          MOVL    SIGNAL_VEC, R0                                ; 1898
                           50      04     A0   D0 00006          MOVL    4(R0), STATUS_CODE                            ;
00000207   8F          50  0C             10   ED 0000A          CMPZV   #16, #12, STATUS_CODE, #519                   ; 1900
                                          12   12 00013          BNEQ    1$                                            ;
                               00000000'  00   DD 00015          PUSHL   LOOP_CHAN                                     ; 1905
```

```
               00000000G  00             01  FB 0001B         CALLS   #1, SYS$DASSGN                          ;
                    FB92  CF             00  FB 00022         CALLS   #0, MOM$FREEBUFFER                      ; 1909
                          50      0918   8F  3C 00027 1$:     MOVZWL  #2328, R0                               ; 1913
                                             04 0002C         RET                                             ; 1915

; Routine Size:  45 bytes,    Routine Base:  $CODE$ + 0D0A
; 1937           1916  1
; 1938           1917  1 END
; 1939           1918  1
; 1940           1919  0 ELUDOM



                                                                            .EXTRN  LIB$SIGNAL

;                                PSECT SUMMARY
;
;       Name                  Bytes                     Attributes
;
;    $OWN$                           652  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;    $PLIT$                          224  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;    $CODE$                         3383  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                           Library Statistics
;
;                                              -------- Symbols --------        Pages       Processing
;       File                                   Total    Loaded   Percent       Mapped      Time
;
;    _$255$DUA28:[MOM.OBJ]MOMLIB.L32;1            194        61        31          21       00:00.1
;    _$255$DUA28:[SHRLIB]NMALIBRY.L32;1           887        32         3          47       00:00.2
;    _$255$DUA28:[SHRLIB]EVCDEF.L32;1             213         3         1          15       00:00.1
;    _$255$DUA28:[SHRLIB]NET.L32;1               1279        26         2          63       00:00.3
;    _$255$DUA28:[SYSLIB]LIB.L32;1              18619        37         0        1000       00:06.5



;                              COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:MOMTEST/OBJ=OBJ$:MOMTEST MSRC$:MOMTEST/UPDATE=(ENH$:MOMTEST)

; Size:           3383 code + 876 data bytes
; Run Time:          01:02.7
; Elapsed Time:      02:13.4
; Lines/CPU Min:      1835
; Lexemes/CPU-Min:  15694
; Memory Used:   257 pages
; Compilation Complete
```

0239 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY